Anomali AQL Search

User Guide

July 23, 2026

Copyright Notice

Copyright © 2026 Anomali, Inc. All rights reserved.

Anomali is a registered trademark, ThreatStream is a registered service mark, and Optic, Integrator, STAXX, Anomali Enterprise, Anomali Match, Anomali Security Analytics, Anomali Lens, and Anomali Copilot are trademarks of Anomali, Inc.

All other brands, products, and company names used herein may be trademarks of their respective owners.

Support

Support Portal	https://support.anomali.com
Email	support@anomali.com
Phone	+1 844-4-THREATS

Documentation Updates

Date	Description
Jul 23, 2026	Updated information about the OCSF Field Summary Panel that it now shows all OCSF fields for the relevant event categories when querying OCSF-normalized data.
Jul 8, 2026	Updated documentation for streamstats operator with new support for dc aggregation, and support for isnull and isnotnull in the Basic Mode for Search.
Jun 24, 2026	Added OCSF Field Panel information that becomes visible when "Exploring Search Results and Refining Queries" on page 72.
Jun 10, 2026	Added support for OCSF schema views that allow access to data fields across OCSF-compliant data sources, regardless of vendor.

Date	Description
May 28, 2026	Added information for OCSF support for Copilot-based searches, correlation tables, full-text indexing, turbosearch, and nested objects in lookup tables, and updates to the rex, transaction, and iplocation operators.
Apr 29, 2026	Updated information for the "whois" on page 712 operator, OCSF support in Search Tabs, "OCSF Schema" on page 768 update for Version 3 fields, and custom suffixes in Report filenames.
Apr 15, 2026	Added information about "Resource Tags" on page 146, OCSF updates for operators, GeoMap visualization updates, DecisionTreeRegressor and RandomForestClassifier updates, calc shannon_entropy function, and timechart operator update.
Mar 31, 2026	Added information about the "OCSF Schema Overview" on page 39, asset and investigation_elements lookup tables, and updates to the MITRE ATT&CK Heatmap visualization.

Date	Description
Mar 4, 2026	Added information about the Google Threat Intel ("gti" on page 577) operator and updates to "fit ARIMA" on page 542.
Feb 19, 2026	Added Reports.
Feb 19, 2026	Updated the "CrowdStrike Operators " on page 508section.
Feb 18, 2026	Added "fit OrdinalEncoder" on page 566, "ml_model_summary" on page 614, "fit StandardScaler" on page 572, and "fit DensityFunction" on page 552.
Dec 10, 2025	Added information about role-based access control information for "Using Lookup Tables" on page 125 and "Using Macros" on page 198. Also added information about the "fit IsolationForest" on page 556 machine learning operator.
Dec 9, 2025	Updated "kmeans" on page 592 and "fit DecisionTreeRegressor" on page 548.

Date	Description
Dec 3, 2025	Added information about the new `ioc_type_v2` and `itype_v2` fields in the observables lookup table. See "observables" on page 1298 for more information. Added information about extracted fields. See Extracted Fields for more information.
Nov 17, 2025	Added information about the bubble chart visualization type.
Nov 5, 2025	Enhanced an example in the "mvexpand" on page 619 operator document.

Send Documentation Feedback

If you have comments about this document, you can contact the documentation in these ways:

- Click contact the documentation team to send an email. If you have an email client configured on this system, an email window will open with the following information in the subject line:

  Feedback on User Guide (AQL Search)

- Send your feedback to docs@anomali.com.

Thank you for your feedback!

CONTENTS

Documentation Updates 3

Send Documentation Feedback 7

Chapter 1: Anomali Search Overview 19

- Anomali Query Language 19
- Search 19
- Dashboards 19
- Lookup Tables 20
- Use Cases 20

Chapter 2: Anomali Search 21

- Types of Anomali Search 24
- Choosing the Right Search Method to Optimize AVC Usage 36
- Why Raw Message Scans (message contains) Are Expensive 37
- Event Log Search — Parsed Fields 37
- Turbosearch — High-Speed Indexed Search 38
- Full-Text Search — When a Value May Not Be in an Indexed Field 38
- Table / Lookup Search — Lowest-Cost Option for Enriched Data 38
- Searching Archived Data Beyond 90 Days 39
- OCSF Schema Overview 39
- Using Search Modes 47
- Understanding the Search Bar 58
- Search with Anomali Copilot 68
- Exploring Search Results and Refining Queries 72
- Guidelines for Search 87
  - Search Throttling 87
  - Users, Roles, and Role-based Search 87
  - Recommended actions 88

Saved Searches 88
Working with Visualizations 101
Modifying Search Settings 109

Chapter 3: Role-Based Access Control for Schemas 110
About Role-Based Access Control 110
Schema Filter in Roles 111
Configuring a Schema Filter for a Role 112
How the Schema Filter Works 113
Schema Filter Scope Across Search Features 116
Schema-RBAC Enforcement 117

Chapter 4: Full-Text Log Indexing 118
Key Technical Guidelines 118
How to Use 119

Chapter 5: Search Data Retention Policy 121
Searching Archived Data 121

Chapter 6: Anomali Virtual Compute 122
Operational Scope 122
Target audience 122
Monitoring usage 122
Best Practices for Managing Usage 122
Monitor your AVC usage 123

Chapter 7: Using Lookup Tables 125
User Sharing and Permissions 130
Anomali System Lookup Tables 131
Creating Custom Lookup Tables 136
Managing Custom Lookup Tables 140

Chapter 8: Resource Tags 146
User Sharing and Permissions 148
Audit Logging Resource Tags 150

Creating a Resource Tag 150
Renaming and Merging Resource Tags 153
Deleting Resource Tags 157
Audit Logging for Resource Tags 158

Chapter 9: Reports 162
Creating a Report 165
Applying Tags to Reports 169
Managing Report Tags 171

Chapter 10: Views 175
Benefits 175
Overview 176
Create a View 180
Running a View 185
Update a View 188
Filtering Views by Resource Tag 190
Bulk Editing Tags and Permissions for Views 191
Clone a View 193
Delete a View 193

Chapter 11: Job Tracking 195
Monitor background jobs 195

Chapter 12: Using Macros 198
Overview 199
User Sharing and Permissions 202
Creating a Macro 203
Managing Custom Macros 205
Using Custom Macros in Search 208
Viewing Custom Macros 210
Using ThreatStream Macros in Search 210
Available Macros 211

Chapter 13: Federated Search 249

How federated search works 249
Best Practices 250
Examples 250

Chapter 14: Forensic and Retrospective Search 251

Chapter 15: About AQL Visualization Types 252
About the Bar Chart Visualization Type 253
About Base64 Visualization Type 266
About the Bubble Chart Visualization Type 270
About the Copilot Widget 273
Key features 274
Use cases 274
Using Copilot in Dashboards 276
About the Decision Tree Visualization Type 277
About the Heatmap Visualization Type 280
About the Gauge Visualization Type 288
About the Geo Map Visualization Type 298
About the Gantt Time Series Visualization Type 301
About the Linear Regression Visualization Type 310
About the MITRE ATT&CK Flow Visualization Type 313
Panel Options 314
Exporting Attack Flows 316
About the MITRE ATT&CK Heatmap Visualization Type 316
Panel Options 318
About the Pie Chart Visualization Type 327
About the Sankey Diagram Visualization Type 337
About the Stat Visualization Type 339
About the Table Visualization Type 349
About the Text Visualization Type 363
Support for Variables 370
About the Timeline Visualization Type 373
Support for Variables 381
About the TimeSeries Visualization Type 384
About the Wordcloud Visualization Type 394

Chapter 16: Anomali Query Language Reference 403
AQL Cheat Sheet 403
Filter Search 403
Select Results Table Columns 404
Aggregate Rows 405
Aggregation over Timespan 405
Limit Results 405
Sort Results Table 406
Calculate Fields 406
AQL Operators by Use Case 407
Operators with Functions 409
AQL Best Practices 409
Use superapi Efficiently 409
Use OCSF Index to Reduce Parsing Overhead 410
Resolving error: "To optimize your experience and reduce your AVC usage, Anomali recommends reducing the time range to 24h to proceed." 411
Searching across different time ranges 413
Use rex Operator and json_extract Efficiently 414
Use array_agg Operator Effectively 414
Use the dedup Operator Effectively 414
Stagger Scheduled Tasks 415
Use Definitive Operators 415
Place Limits Last 415
Use Smallest Possible Time Range 416
Specify Targeted Search Table 416
Prioritize Predicates 416
Use Normalized Fields 416
Dotted notation 417
Subsearch 422
abuseipdb 428
accum 429
addcoltotals 429
addinfo 431
addinv 432
addsums 434
addtag 435

aggr ... 436
anomalousvalue ... 442
anomalydetection ... 446
appendcols ... 448
appendtable ... 449
apply ... 451
Support for explicit arguments ... 451
autoregress ... 457
bin ... 459
bucketdir ... 460
calc ... 461
chart ... 497
cluster ... 499
cofilter ... 501
concurrency ... 503
contingency (alias: ctable) ... 504
convert ... 505
creinv ... 507
CrowdStrike Operators ... 508
cs_query_alerts ... 509
cs_get_alert_entities ... 513
cs_query_device ... 514
cs_get_system_info ... 516
cs_get_process_detail ... 516
cs_hunt_ip ... 517
cs_hunt_domain ... 519
cs_hunt_file ... 520
cs_list_users ... 521
curl ... 522
dedup ... 523
deletemodel ... 524
domaintools ... 525
erex ... 525
evalrule ... 527

eventstats 530
extract 532
fields 533
fieldsummary 535
filldown 538
fillnull 538
fit 539
- fit ARIMA 542
- fit birch 546
- fit DBSCAN 547
- fit DecisionTreeRegressor 548
- fit DensityFunction 552
- fit IsolationForest 556
- fit LinearRegression 559
- fit LocalOutlierFactor 561
- fit LogisticRegression 563
- fit OrdinalEncoder 566
- fit RandomForest 568
- fit StandardScaler 572

foreach 574
fromjson 576
gti 577
head 579
help 579
history 580
import_to_ts 582
inlist 584
iplocation 585
jira 587
join 588
kmeans 592
like 595
limit 597
listmodels 598

listinv 598
listopinv 599
loadjob 600
lookup 601
lstintdest 604
lstperftag 605
makejson 606
makemv 607
makeresults 609
mcatalog 611
mdti_ssl 613
metasearch 614
ml_model_summary 614
multikv 616
mvcombine 618
mvexpand 619
mvreverse 620
nomv 621
openports 622
outlier 622
outputtext 625
predict 625
rangemap 630
rare 632
recordedfuture 633
reltime 634
replace 635
return 636
reverse 638
rex 639
rmvtag 641
sandbox 642

savedsearch 643
score 644
scrub 647
selfjoin 649
set 650
setfields 651
shodan_database 652
sort 653
snow 654
snowevent 655
snoweventstream 656
snowincident 658
spath 661
streamstats 663
superapi 666
Before you begin 667
tail 676
timechart 677
timetrend 682
tojson 683
top 686
transaction 687
transpose 689
turbosearch 690
tsquery 696
uniq 698
virustotalv3 699
where 700
whois 712
xmlkv 715
xmlunescape 716
xpath 717

xyseries 718
yieldtable 719

Reference Information 722
actor_associations 722
asset 725
investigation 733
investigation_elements 735
eventlog 746
OCSF Schema 768
OCSF Application Activity Schema 769
OCSF Discovery Schema 833
OCSF Findings Schema 886
OCSF Identity & Access Management Schema 994
OCSF Network Activity Schema 1062
OCSF System Activity Schema 1163
OCSF Correlation Tables 1225
dgamatch_ocsf 1226
iocmatch_ocsf 1232
tmmatch_ocsf 1240
consolidated_actor_descriptions_ng 1241
consolidated_actor_observables_ng 1245
consolidated_actor_profiles_ng 1250
Appendix B: dgamatch 1255
Appendix C: ignite 1267
Appendix D: iocmatch 1281
Appendix D: observables 1298
Appendix E: tmmatch 1305
Appendix E: threat_models 1306
Appendix F: Reference: Date Time Format Code 1320
Using the Search RESTful API 1321
About the API 1321
Create a New Search 1321
Get Search Status 1324
Get Search Results 1328

Chapter H: Using Lookup Tables (Classic UI) 1334
Anomali System Lookup Tables (Classic UI) 1336
Creating Custom Lookup Tables (Classic UI) 1339
Managing Custom Lookup Tables (Classic UI) 1342
Chapter I: Using Macros (Classic UI) 1348
Creating a Macro (Classic UI) 1349
Managing Custom Macros (Classic UI) 1351
Using Custom Macros in Search (Classic UI) 1353
Viewing Custom Macros (Classic UI) 1355
Using ThreatStream Macros in Search (Classic UI) 1356
Available Macros 1357

Chapter 1: Anomali Search Overview

Anomali Search is a data analytics platform designed to enable important cybersecurity practices.

Anomali Query Language

The Anomali Query Language (AQL) supports:

- Evaluation of standard Boolean expressions to filter data
- Aggregation functions
- Calculation and transformation functions
- Timetrends
- Enrichment from ThreatStream intelligence sources and external sources, such as GeoIP data sets
- Training machine learning models on event log data

Search

Anomali Search can parse natural language requests. The Search user interface also features typeahead suggestions and click-to-add filters from results to help you get started with AQL.

See "Anomali Search" on page 21.

Dashboards

Anomali Search dashboards are powered by AQL searches and provide a useful set of visualization options.

See Dashboards.

Lookup Tables

Anomali supports several lookup tables that let you map your event log data to Anomali's schema, perform different types of IOC correlation, and enable further analysis. See Lookup Tables for more information.

Use Cases

- Threat hunting

  Use natural language or precise AQL expressions to explore your event logs and correlate event data with ThreatStream intelligence and enrichment.

- Automatic detection of indicators of compromise (IOC)

  Upon ingestion, event logs are correlated with ThreatStream intelligence to produce the IOC Match system table and with high fidelity intelligence identifying endpoint filehash IOCs to produce the Ignite Match system table.

- User, entity, and indicator behavior analytics (UEIBA)

  Leverage the IOC Match and Ignite Match system tables or use the raw event log to train machine learning models. Anomali supports a rich set of algorithms including ARIMA (Autoregressive Integrated Moving Average, DBSCAN (Density-Based Spatial Clustering of Applications with Noise), k-means clustering, LinearRegression, LocalOutlierFactor, LogisticRegression, StandardScaler, and more. Refer to "AQL Operators by Use Case" on page 407 for an index of UEIBA operators.

- Security orchestration, automation, and response (SOAR)

  Use alerts to add tags to ThreatStream observables or to send AQL results to a preferred destination. You can also use AQL operators to get or post columnar data. Refer "AQL Operators by Use Case" on page 407 for an index of SOAR operators.

Chapter 2: Anomali Search

Anomali Search enables analysts to scan, correlate, and analyze large volumes of security telemetry and threat intelligence. Using Search, analysts query event logs, lookup tables, and views to investigate activity and detect potential security incidents.

Search supports querying data using two schema standards: the eventlog schema and the Open Cybersecurity Schema Framework (OCSF) schema. Both these schemas support normalized event data as well as unstructured raw log messages. Fields can be automatically extracted from raw messages (such as JSON, CEF, or key-value logs) and queried using dotted notation. This schema-on-the-fly capability allows you to rapidly use data in downstream applications such as Visualizations or Anomali Dashboards, without requiring prior field normalization.

Organization administrators can control access to both eventlog and OCSF data across all Search features by assigning roles with targeted filters, ensuring each user sees only the data their function requires. See "Role-Based Access Control for Schemas" on page 110 for more information.

Search queries are written using the Anomali Query Language (AQL), generated automatically using the Basic query builder, or natural-language prompts through Anomali Copilot. These queries allow analysts to filter, aggregate, and analyze large data volumes to support threat hunting and investigations.

Additionally, Anomali offers Anomali Virtual Compute, a computing framework that enables you to measure your usage of virtualized infrastructure whenever you launch a search.

Search also supports federated searches that use the Managed Security Service Provider to enable querying across multiple managed organizations at once, while keeping each customer's data isolated and clearly identified in results. See Federated Search to understand how Anomali ensures multi-tenancy.

Accessing the Search Interface

Click **Search** ( ) in the left navigation panel to go to the Search home page.

Search Types

Anomali Search supports multiple search types, each optimized for different use cases. See "Types of Anomali Search" on page 24 for guidance on choosing the right search type.

Search Modes

Search supports two modes of constructing a search query:

- **AQL** mode: Manually construct queries using AQL operators, filters, aggregations, and regular expressions.
- **Basic** mode: Build queries by selecting components of an AQL query without requiring knowledge of AQL syntax, simplifying query construction.

See "Using Search Modes" on page 47 for more information to understand how these modes enhance your search experience.

OCSF Schema

Anomali Search supports Version 1.2.0 of the Open Cybersecurity Schema Framework (OCSF), an industry-standard schema for normalizing security event data across diverse log sources. OCSF data coexists with the standard `eventlog` schema — to query OCSF events, prefix your search query with `ocsf`.

OCSF data fields use dotted notation to represent nested objects, for example, `src_endpoint.ip` or `actor.user.name`.

See "OCSF Schema Overview" on page 39 for the full field reference, supported categories, and query examples.

Understanding the Search Bar

Use the Search bar to enter AQL queries or natural language prompts and refine the generated query before running it. See "Understanding the Search Bar" on page 58 for more information.

Exploring Results and Refining Queries

Review search results, inspect field values, and apply filters or follow-up actions to narrow your search results and focus your investigation. See "Exploring Search Results and Refining Queries" on page 72 for more information.

OCSF Field Panel

The OCSF Field Panel is a sidebar in Event Search that lists OCSF Version 1.2.0 schema fields by category, shows how many events in your current result set have each field populated, and lets you add include or exclude filters to your active query by clicking a field, without typing field names manually in the search bar. See "OCSF Field Panel" on page 80 for more information.

Searching with Anomali Copilot

Anomali Copilot lets you convert natural-language prompts into valid AQL queries that you can review, edit, and run from the Search page. See "Search with Anomali Copilot" on page 68 for more information.

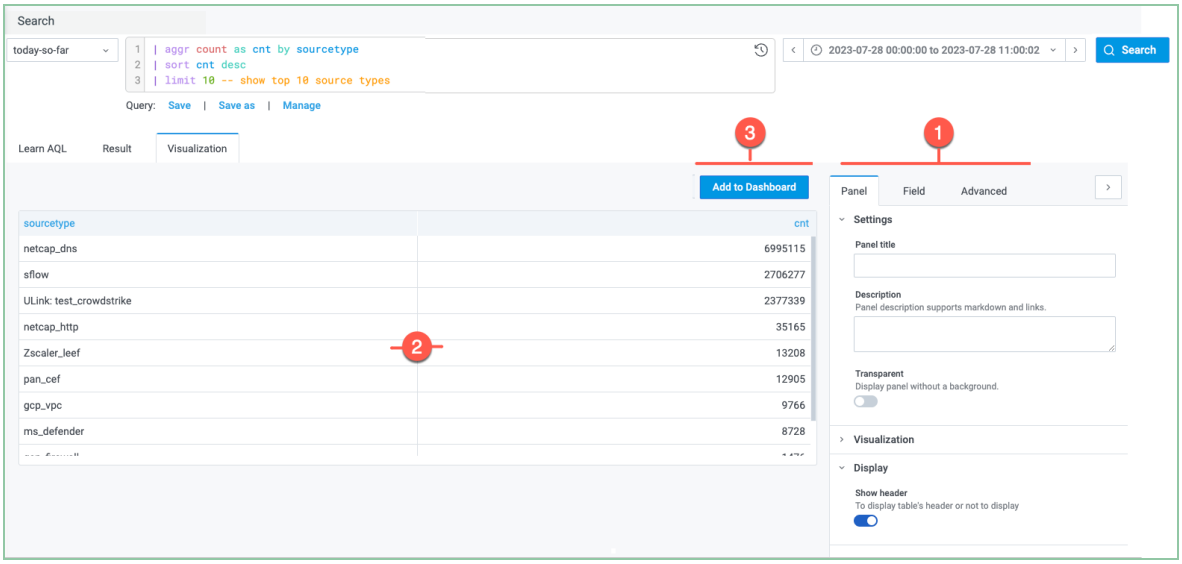
Using Saved Searches

The Event Search page also includes the Saved Search utility that lets you save the current query to a configuration you can reuse in the future. See "Saved Searches" on page 88 for more information.

Searching with Visualizations

After refining a query, navigate to the **Visualization** tab to get a visual snapshot of your search results. Select a visualization appropriate for your use case, customize its properties, and save it to your dashboard.

See "Working with Visualizations" on page 101 for more information on all the visualizations Anomali Search supports.



Tagging Anomali Resources

Search also allows you to tag different Anomali resources such as lookup tables, views, macros, reports, saved searches, and dashboards, leading to a shared, organization-wide tagging system that eases the process of organizing and locating them. See "Resource Tags" on page 146 for more information.

Modifying Search Settings

Search also allows you to toggle between light and dark mode themes for a better user experience. See "Modifying Search Settings" on page 109 for more information.

Types of Anomali Search

Anomali Search offers four search types, each optimized for different use cases:

- Field-based Search: eventlog and OCSF schema queries using parsed and normalized fields.
- TurboSearch: high-speed indexed lookups for exact-value searches and IOC sweeps.
- Full-Text Search: keyword search on raw log messages using full-text indexing.
- Table / Lookup Search: queries against pre-enriched or correlated datasets.

See "Choosing the Right Search Method to Optimize AVC Usage" on page 36 to determine the search method that has the most optimal AVC usage for your application.

Type	What it is	When to use	Key guidance & best practices
Field-based Search **Default**	eventlog Search (**default**) • Searches normalized security telemetry in the `eventlog` dataset. • Supports structured and extracted fields. • Default if no dataset specified.	When you know the schema(for example, `src_ip`, `user`, `event_name`).	**Indexing Latency:** 10-30 seconds after ingestion. **Filter strategy** • Lead with sourcetype as the primary constraint. • Use parsed or normalized fields. • Do not use `message contains` as a substitute for structured fields. **Prioritize parsed fields** Parsed fields are significantly more efficient than raw message searches. Use structured field searches wherever possible. ✗ High AVC: `| where raw_message contains_ci ("John")` ✓ Recommended: `user = "John"` **Avoid `message` field on large ranges** If you must use `message`, restrict the query to a 24-hour window and always pair it with a `sourcetype` filter. **Operator order** Place `where` clauses before `calc`, `rex`, and

Type	What it is	When to use	Key guidance & best practices
			`json_extract` to skip unnecessary processing. Place `limit` last to avoid non-deterministic results. **Use definitive operators** Prefer `=` and `!=` over `contains` or `contains_ci` for accurate, efficient filtering. **Use normalized fields** Check if a JSON field has a normalized column in the eventlog schema and use that directly instead of `json_extract(message, ...)`. **Avoid wildcards on sourcetype** ✗ `sourcetype contains '%'` ✓ `sourcetype = 'defender'` **Stagger scheduled tasks** Distribute alerts and reports across a time window (for example, 2–3 AM) instead of triggering them simultaneously..

Type	What it is	When to use	Key guidance & best practices
	OCSF Search • Stores event telemetry normalized to the OCSF Version 1.2 schema. • Uses nested object notation. • Requires the case-insensitive keyword prefix `"ocsf"`	• When your data is normalized to the OCSF Version 1.2 schema. • When querying across multiple vendors or log sources using standardized OCSF field names. • When filtering by OCSF event category (for example, Network Activity, Discovery). • When you need OCSF dotted-notation fields such as `src_endpoint.ip` or `actor.user.name`.	**Indexing Latency:** 10-30 seconds after ingestion. **Filter strategy** • Lead with `category_name` as the primary constraint (equivalent of `sourcetype` in eventlog). • Use OCSF dotted-notation fields. Do not use flat eventlog field names in OCSF queries. • Do not use `message contains` as a substitute for structured OCSF fields. **Prioritize nested fields** OCSF fields are pre-indexed. Querying them is as efficient as querying parsed eventlog fields. ✗ High AVC: `| where message contains_ci ("10.0.0.1")` ✓ Recommended: `ocsf | where category_name = "Network Activity" | where src_endpoint.ip = "10.0.0.1"` **Operator order** Place `where` clauses before `calc`, `rex`, and

Type	What it is	When to use	Key guidance & best practices
		Filter strategy • Lead with `category_name` as the primary constraint (equivalent of `sourcetype` in eventlog). • Use OCSF dotted-notation fields. Do not use flat eventlog field names in OCSF queries. • Do not use `message contains` as a substitute for structured OCSF fields. **Prioritize nested fields** OCSF fields are pre-indexed. Querying them is as efficient as querying parsed eventlog fields. ✗ High AVC: `| where message contains_ci ("10.0.0.1")` ✓	

Type	What it is	When to use	Key guidance & best practices
		Recommended: `ocsf | where category_name = "Network Activity" | where src_endpoint.ip = "10.0.0.1"` **Operator order** Place `where` clauses before `calc`, `rex`, and `json_extract` to skip unnecessary processing. Place `limit` last to avoid non-deterministic results. **Use definitive operators** Prefer `=` and `!=` over `contains` or `contains_ci` for accurate, efficient filtering. **Stagger scheduled tasks** Distribute alerts and reports across a time window (for example, 2–3 AM) instead of	

Type	What it is	When to use	Key guidance & best practices
		triggering them simultaneously.	

Type	What it is	When to use	Key guidance & best practices
TurboSearch **Accelerated** (Field-based and keyword search on specific fields)	Optimized search, designed for high-speed lookups on specific indexed fields.	• Searching for **exact values** in known indexed fields. • Multi-value IOC sweeps. • For Forensic and retrospective searches on large datasets, threat hunting.	**Indexing Latency:** 10 minutes after ingestion. **Query Type** Can specify a keyword or field-based search queries. Values are searched only in the indexed fields as described below. **Indexed fields** `account_id`, `dest`, `dest_ip`, `dvc_name`, `file_hash`, `file_hash2`, `file_hash3`, `file_name`, `host`, `http_referrer`, `message_id`, `query`, `src`, `src_ip`, `url`, `user` **Match Behavior** - Prefix* and exact match are supported. - Wildcard, suffix, and “contains” (i.e., partial matches within a string) are not supported. **Time Range Guidance** Use for any time range **AVC Impact** Use Turbosearch as the initial operator. Combine with indexed fields and parsed field filters for low AVC. **Examples**

Type	What it is	When to use	Key guidance & best practices

			`|turbosearch "1.2.3.4"`
			`|turbosearch src_ip = '1.2.3.4'`
			`|turbosearch sender='sender_example@example.com'`
			`|turbosearch "https://www.example-url.com/feed"`
			`|turbosearch "tech@acmecorp.com", "examplefilehash" | aggr count (sourcetype) as srctype_count`

Type	What it is	When to use	Key guidance & best practices
Full-Text Search **Keyword**	Keyword search on raw log messages using full-text indexing. Faster and lower AVC cost than searching the raw message (using `message contains`)	• You do NOT know field names • Searching for exact keywords (emails, strings, IDs). • **Only applicable to eventlog.**	**Indexing Latency:** 10 minutes after ingestion. **Syntax requirements** Keyword must be at the start of the query, wrapped in double quotes. Search is case-insensitive. ✓ "john.smith@company.com" ✓ `"john.doe@company.com" | where sourcetype="email_logs"` **Tokenization** • Valid tokenization characters: letters, digits, dot `(.)`, dashes `(-)`, underscore `(_)`. • Delimiters: spaces, punctuation, and special characters. **Match Behavior** • Exact matches only. • Does not support wildcards, partial matches, or substrings, including for sourcetype. **Boolean Support**

Type	What it is	When to use	Key guidance & best practices
			AND and OR supported but not NOT **Case Handling** Search is case-insensitive. For example, "Error" and "error" are treated the same. **Time Range Guidance** Use for any time range **Turbosearch vs Full-text** More granular than Turbosearch keyword search as Turbosearch matches full values while full-text matches tokens. Example: For user = john.doe@company.com Turbosearch: user = "john.doe@company.com" will match user = "john" will not match Full-text: "john" will match "doe" will match "company.com" will match due to tokenization rules

Type	What it is	When to use	Key guidance & best practices
Table / Lookup Search **Enriched**	Search against pre-enriched or correlated datasets: `iocmatch`, `threat_models`, `audit`, and so on. Reduces scanned dataset versus default eventlog. See "Using Lookup Tables" on page 125 for more information.	• When working with pre-enriched data or correlated threat intelligence. • When you want to narrow the dataset away from the full `eventlog`.	**Indexing Latency:** 15 minutes after ingestion System tables are updated every 10 minutes **Specify the table explicitly** Name the target table at the start of the query to avoid a full `eventlog` scan. ✓ `iocmatch | where action in ('allow', 'accept')` ✗ (higher AVC) eventlog | where action in ('allow', 'accept') | where src_ip = "1.2.3.4 **array_agg limits** The `array_agg` operator is capped at 50 distinct values. Use low-cardinality fields, filter data before aggregating, or use `group by` for high-cardinality fields. **dedup guidance** Use `dedup` on small, pre-filtered datasets only. Never apply it to the `message` field — each raw event is held in memory, hurting performance.

Type	What it is	When to use	Key guidance & best practices
Forensic / Retrospective Search	By default, Anomali Search retains event data in hot buckets for 90 days, after which it moves to archive buckets based on license retention terms. Searches run only on hot data by default to control AVC usage. Access to archived data requires approval via a support ticket, after which your account can be enabled for extended searches. Once enabled, use eventlog, Turbosearch, or full-text search to search.		

Choosing the Right Search Method to Optimize AVC Usage

Every search you run in Anomali Search consumes AVC units. The method you choose, the time range you set, and the data source you target all determine how much work the system must do — and therefore how many AVC units your query costs.

Anomali Search provides four search methods. Each serves a different purpose and carries a different AVC cost. Choosing the right one for the task at hand is the single most effective way to control resource consumption.

Search Method	AVC Cost
Field-based Search	eventlog (*Default*) ▼ Low — uses parsed and indexed fields; fastest for hot-data queries when the schema is known. OCSF Search ▼ Low — equivalent cost to Event Log parsed-field queries; uses the `ocsf` table and dotted-notation fields instead of `eventlog`; queries must specify a `category_name` constraint.
Turbosearch *(Accelerated)*	▼ Low–medium — indexed field lookups are fast and efficient; recommended over `message contains` for any exact-value search on indexed fields.

Search Method	AVC Cost
Full-Text Search *(Keyword)*	▼ Low–medium — cheaper than `message contains` for exact token matches; no wildcards or partial matches — exact match only.
Table / Lookup Search *(Enriched)*	▼ Very low — queries a small, pre-correlated dataset instead of the full `eventlog`; requires the table name to be specified explicitly at the start of the query.

Why Raw Message Scans (message contains) Are Expensive

When Anomali Search receives a query that uses `| where message contains_ci`, it must read every raw log line stored in the specified time range and check whether the search string appears anywhere inside it. For a 24-hour window with high-volume data, that can mean millions of records scanned. For a 90-day window, the cost multiplies accordingly.

Event Log Search — Parsed Fields

Parsed fields eliminate the raw-scan problem. During ingestion, Anomali extracts known values from your logs — usernames, IP addresses, hostnames, event names — and stores them in structured columns. A query on a parsed field reads only that column, not the full raw message. The system returns results faster and consumes a fraction of the AVC units.

> **Tip:** If you are unsure which parsed field to use, start with Turbosearch or full-text search as described in the sections below.

Use Case	Avoid	Recommended
Finding a user	`| where message contains_ci ("John")`	`user = "John"`
Finding by IP address	`| where message contains_ci ("192.168.1.10")`	`src_ip = "192.168.1.10"`

Use Case	Avoid	Recommended
Finding a file hash	`| where message contains_ci ("3b4c5a48c45a4c3b56a2")`	`file_hash = "3b4c5a48c45a4c3b56a2"`

Turbosearch — High-Speed Indexed Search

Turbosearch is a dedicated high-speed search operator that queries a set of pre-built indexed fields directly, bypassing the need to scan raw log records. Because it operates on an index rather than the full message text, it returns results significantly faster and at lower AVC cost than equivalent raw-message or standard parsed-field queries — especially over larger time windows.

Full-Text Search — When a Value May Not Be in an Indexed Field

Full-text search uses a pre-built index on the raw message, so it finds exact token matches without reading every record from scratch. This makes it significantly cheaper than a raw message scan.

Use full-text search when you need to search the raw message for an exact value and that value does not have a corresponding parsed field in the `eventlog` schema. The classic use cases are email addresses, URLs, or unique identifiers that are not extracted as named columns during ingestion.

Use Case	Avoid	Recommended
Searching for an email address	`| where message contains_ci ("john.doe@company.com")`	`"john.doe@company.com" | where sourcetype="email_logs"`

Table / Lookup Search — Lowest-Cost Option for Enriched Data

Anomali maintains pre-correlated datasets — such as `iocmatch`, `ignite`, and `threat_models` — that contain the results of correlation work already performed by the platform. Querying these tables is significantly cheaper because the scanned dataset is a fraction of the size of the full event log.

The key to keeping costs low is to name the target table explicitly at the start of the query. If you omit the table name, the query defaults to scanning the full `eventlog`.

Use Case	Avoid (Higher AVC)	Recommended (Lower AVC)
Checking IOC matches	`eventlog | where action in ('allow', 'accept') | where src_ip = "1.2.3.4"`	`iocmatch | where action in ('allow', 'accept')`

Searching Archived Data Beyond 90 Days

By default, Anomali Search queries only hot data — event logs from the past 90 days stored in fast, searchable hot buckets. After 90 days, data automatically moves to archive buckets, where it is retained according to the terms of your license agreement. Archived data is not searchable by default; access is restricted.

To search archived data, submit a support ticket to have your account configured for archive searching. Once enabled, use `eventlog`, Turbosearch, or full-text search as appropriate.

OCSF Schema Overview

Anomali Search supports Version 1.2 of the Open Cybersecurity Schema Framework (OCSF) schema, an industry-standard schema for normalizing security event data. OCSF provides a consistent structure for security telemetry across diverse log sources, enabling interoperability with OCSF-native tools and standardized threat detection workflows.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

OCSF and eventlog: Dual-Schema Architecture

Anomali Search uses a dual-feed architecture where both `eventlog` and `ocsf` tables coexist. Users select which schema to query by specifying a table prefix in their search queries.

Schema	Default Table?	Description	Usage

eventlog	Yes	• Stores event telemetry your organization ingests into Security Analytics. Event fields are normalized to the event log schema, and Search automatically extracts additional fields from raw messages such as JSON.
		• When you search the event log, specify field names from the event log schema, such as `src`, `dst`, `protocol`. See "eventlog " on page 746for more information.
		• `eventlog` is the implicit search table if you do not specify anything. Alternatively, you can optionally specify the table name `eventlog` before the first pipe.
		• The time picker uses event time, which corresponds to the timestamp stored in the event log.
			`eventlog | where src = "10.0.0.1"`
			or omit prefix for default:
			`| where src = "10.0.0.1"`

OCSF	No	• Stores event telemetry normalized to the OCSF Version 1.2 schema. • OCSF fields use nested object notation like `src_endpoint.ip`, or `actor.user.name`. See "OCSF Schema Overview" on the previous page for more information. • To use OCSF fields, specify the case-insensitive keyword `"ocsf"` before the first pipe. For example: `ocsf | where src_endpoint.ip = "10.0.0.1"`	`ocsf | where src_endpoint.ip = "10.0.0.1"`

For the complete eventlog schema reference, see "eventlog " on page 746. For the official OCSF specification, see the official OCSF documentation.

Supported OCSF Categories

Anomali Search supports the following OCSF Version 1.2 categories:

OCSF Category	Description
System Activity	Events related to operating system activities, including process creation, file system operations, kernel events, and system service activities.
Findings	Security findings, alerts, and detections from security tools such as EDR, SIEM, vulnerability scanners, and threat intelligence platforms.
Identity and Access Management	Authentication events, authorization activities, user account management, privilege escalation, and identity-related security events.
Network Activity	Network traffic events including connections, flows, HTTP requests, DNS queries, firewall logs, and network protocol activities.
Discovery	Reconnaissance and discovery activities, including device enumeration, service discovery, network scanning, and information gathering events.
Application Activity	Application-level events such as web application logs, API calls, cloud service activities, and software-specific security events.

To query events from a specific OCSF category, use the `category_name` field. For example:

```
ocsf | where category_name = "Network Activity"
```

Supported OCSF Classes for Anomali Copilot

Anomali Copilot supports the following OCSF classes:

OCSF Category	Description
API Activity	API call events capturing request and response details, including REST endpoints, methods, status codes, and resource identifiers for cloud and application service interactions.
Account Change	User account lifecycle events including account creation, deletion, enabling, disabling, password changes, and attribute modifications.
Authentication	Authentication and session events including login attempts, logon successes and failures, multi-factor authentication, and single sign-on activities.
Email Activity	Email handling events covering message sending, receiving, delivery, forwarding, and filtering activities, including attachment and email metadata.
Entity Management	Lifecycle events for managed entities such as configurations, policies, certificates, and organizational objects, including creation, modification, and deletion.
Filesystem Activity	File system operation events including file creation, deletion, modification, access, and attribute changes across local and network storage.
Group Management	Group lifecycle events covering group creation, deletion, and modification, including membership additions and removals.
HTTP Activity	Web traffic events capturing HTTP and HTTPS request and response details, including URLs, methods, status codes, user agents, and payload metadata.
Network Activity	Network connection and traffic events including TCP/UDP sessions, flows, firewall logs, and network protocol activities between endpoints.
Process Activity	Process lifecycle events including process launch, termination, injection, and other activities that describe process execution and behavior on a host.

Using OCSF in Search

To query OCSF data, prefix your search query with `ocsf` followed by a pipe (`|`) and your AQL operators.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Basic OCSF Query Syntax

```
ocsf | where <field> = <value>
```

OCSF Field Notation

OCSF uses dotted notation to access nested object fields. Use the dot (`.`) operator to traverse through nested structures.

The following table presents a few examples of how eventlog fields map to their equivalent OCSF fields:

eventlog Field	OCSF Equivalent
`src`	`src_endpoint.ip`
`dst`	`dst_endpoint.ip`
`src_port`	`src_endpoint.port`
`dest_port`	`dst_endpoint.port`
`user`	`actor.user.name`
`protocol`	`connection_info.protocol_name`

For the complete list of the OCSF Schema fields, see OCSF Schema.

Field Naming Conventions

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

OCSF organizes fields into nested objects representing entities and attributes. Common OCSF object patterns include:

- **Endpoints**: `src_endpoint`, `dst_endpoint` contain IP addresses, ports, hostnames, and network identifiers
- **Actor**: `actor.user`, `actor.process` describe who or what initiated an activity
- **Metadata**: `metadata` contains event timestamps, log sources, and event identifiers
- **Connection Info**: `connection_info` holds network protocol details
- **File**: `file.name`, `file.path`, `file.hash` describe file attributes

For the complete list of the OCSF Schema fields, see OCSF Schema.

Example: Complex Nested Structure

The following query demonstrates accessing deeply nested OCSF fields:

```
ocsf | where actor.process.file.name = "powershell.exe" and
actor.user.uid = "S-1-5-21-12345"
```

OCSF Field Panel

The OCSF Field Panel is a sidebar in Event Search that lists OCSF Version 1.2.0 schema fields by category, shows how many events in your current result set have each field populated, and lets you add include or exclude filters to your active query by clicking a field, without typing field names manually in the search bar. See "OCSF Field Panel" on page 80 for more information.

OCSF for Role-Based Access Control

OCSF schema supports Role-Based Access Control, allowing organization administrators to define OCSF filters in user roles to restrict access to specific OCSF categories. See "Role-Based Access Control for Schemas" on page 110 for more information.

Examples

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Example 1: Network Activity Query

Query network connections from a specific source IP address.

eventlog Query	OCSF Query
`eventlog | where src = "10.2.3.128" and dest_port = 443`	`ocsf | where category_name = "Network Activity" and src_endpoint.ip = "10.2.3.128" and dst_endpoint.port = 443`

Example 2: Identity and Access Management

Find authentication attempts for a specific user.

eventlog Query	OCSF Query
`eventlog | where action = "remote user logout" and user = "john.doe"`	`ocsf | where category_name = "Identity & Access Management" | where class_name = "Authentication" | where activity_name = "Logon" and user.name = "john.doe"`

Example 3: System Activity - Process Events

Show process activity events where the initiating (actor) process is a regular file.

```
ocsf | where category_name = "System Activity" and activity_name
= "Process Launch" and actor.process.cmd_line contains
"powershell" and actor.process.cmd_line contains "-enc"
```

Example 4: Aggregation by Source Endpoint

Count network connections by source IP address.

eventlog Query	OCSF Query
`eventlog | aggr count by src`	`ocsf | aggr count by src_endpoint.ip`

Example 5: Findings - High Warning Network Activity

Filter network activity events flagged with a warning-level severity where the outcome or status detail is unknown.

```
ocsf | where category_name = "Network Activity" and severity =
"warning" and status_detail = "Unknown"
```

Example 6: Extracts named and positional groups from a structured string using regex, then returns only the job capture group.

```
ocsf

| limit 1

| calc test='Name: Alice, Age: 30, Job: Engineer'

| rex field=test 'Name: (\w+), Age: (\d+), Job: (?P<job>\w+)'

| fields job
```

Example 7: Retrieves all malicious IP IOC matches, then appends exploit IP IOC matches to produce a combined threat indicator result set for hunting across both IOC types

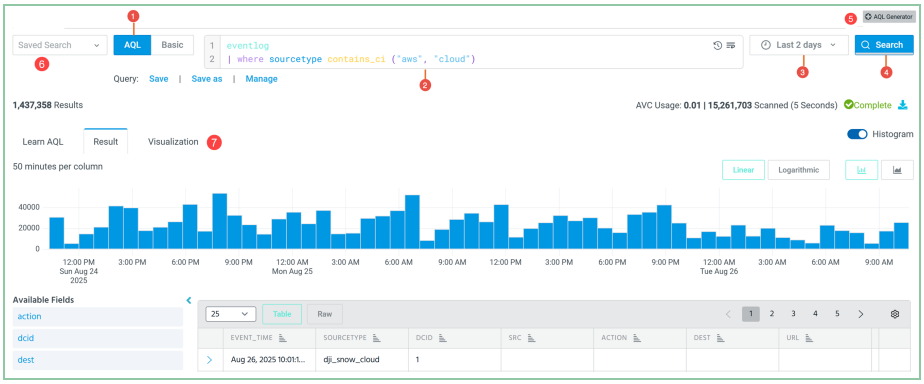
```
iocmatch_ocsf | where itype = 'mal_ip' | appendtable [iocmatch_
ocsf | where itype = 'exploit_ip']
```

Using Search Modes

Search supports two search modes:

- **AQL Query Mode**: In addition to the offerings of the **Basic** Mode, the AQL mode allows you to manually construct queries by using advanced AQL operators, filtering, aggregation, and regular expressions using both the `eventlog` and `ocsf` schemas. See "AQL Query Mode" below for more information.
- **Basic Mode**: The Basic mode lets you construct queries by selecting different components of an AQL query, without prior knowledge of AQL syntax. These components, therefore, lead to a simple and efficient search experience. This mode supports eventlog data, OCSF data, Lookup Tables, as well as Views. See "Basic Mode" on page 52 for more information.

AQL Query Mode



1 2 **AQL Query Mode**: This mode includes all the features of the "Basic Mode" on page 52, as well as covers advanced search functionality using AQL operators, filters, and aggregation functions.

You can start a search in two ways:

- Natural language: Type `--` and then your natural language search. See "Search with Anomali Copilot" on page 68 for more information.

- AQL: Start typing to build an AQL expression. The search bar displays auto-complete suggestions to help you build a valid search query.

See "Understanding the Search Bar" on page 58 to learn more about creating valid queries.

Schema Selection: You can query data using `eventlog` or `ocsf` schemas. If you do not specify a table prefix, `eventlog` is used by default. To use the OCSF schema, prefix your query with the `"ocsf"` case-insensitive keyword.

For example:

eventlog:

```
| where src = "192.168.1.10"
```

OR

```
eventlog | where src = "192.168.1.10"
```

OCSF:

```
ocsf | where src_endpoint.ip =
"192.168.1.10"
```

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

After

3 **Time Range**: Select a relative range, quick range, or specify an absolute time range. The time period filter uses event time, which corresponds to the timestamp in the event log.

Notes:

- In the results table, event time is stored in UTC and adjusted to the web browser time zone.
- The default time range specified in the **My Profile** settings does not apply to Search.

4 Search : Click **Search** to issue the search. Click the icon next to Job to stop the search.

Hover the mouse over the button to view your AVC usage. See Anomali Virtual Compute for more information on how the framework measures your infrastructure usage as per your search.

5 **AQL Generator**: Use natural language to generate an AQL query.

- Click **AQL Generator** then enter a natural language prompt. At this point, the **Search** button changes to **Generate AQL**.
- Once you type your prompt, press **Enter** to generate the corresponding AQL query.
- Click **Search** to run the query.

To generate OCSF queries using AQL Generator, include the case-insensitive keyword `"ocsf"` in your natural language prompt. For example:

```
"Show me OCSF network activity from
192.168.1.10"
```

> **Note:** You must include the case-insensitive keyword `"ocsf"` in your query for AQL Generator to use the OCSF schema. See "Search with Anomali Copilot" on page 68 for more information.

6 **Saved Search**: Review a list of recent saved searches. Select a search to load the query in the search bar and issue the search. See "Saved Searches" on page 88 for more information.

Search History: Review a list of recent searches; select one to load the query in the search bar and issue the search. See "Using Search History" on page 86 for more information.

7 **Focus Tabs**: Toggle between the following tabs:

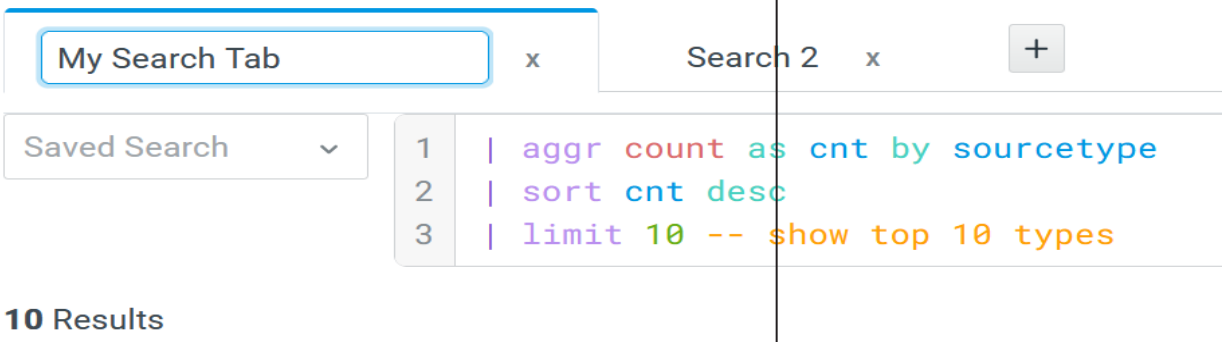
- **Learn AQL**: In the left panel, enter Copilot prompts to convert natural language queries into AQL expressions. See "Using Search Modes" on page 47 for more information.

  Alternatively, in the right panel, you can hover over the example query text and click **Run** to run one of the example AQL expressions.

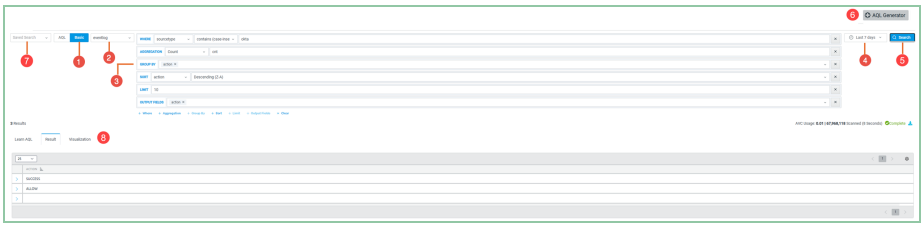
- **Result**: Results appear in a paged table. Review results and use the UI tools to add filters to further refine your search and set fields to be displayed in visualizations. See "Exploring Search Results and Refining Queries" on page 72 for more information.

- **Visualization**: After refining a query, toggle to the **Visualization** tab to select a visualization and configure its properties, prior to add it to a dashboard. See "Using Search Modes" on page 47 for more information.

8 **Search Tabs**: Use one or more search tabs to launch a search query.



Basic Mode



1	**Basic** Mode: Click **Basic** to use the structured query builder. The builder lets you select query components in any order and generate the corresponding AQL query, eliminating the need to manually write the query. • **Multifunction Search**: Launch searches on datasets, lookup tables, or views. • **Ease of use**: Browse event log data or lookup data. • **Increased efficiency**: Browse, filter, and aggregate data without requiring AQL syntax knowledge. • **Flexibility**: Toggle the **AQL Mode** tab to manually construct more detailed queries for specific use cases. **Basic** Mode supports only a limited subset of filtering, aggregation, and sorting options. For more advanced functionality, see "Using Search Modes" on page 47 for more information.

2 **Dataset**: Select a dataset to start a structured search on event log data, a lookup table, or a view. All conditions and operations apply to the selected dataset.

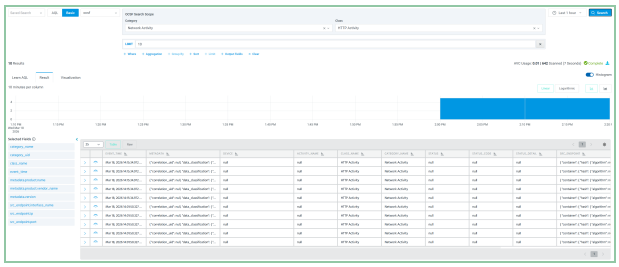
eventlog

Select to query event data using the eventlog schema with standard field names.

OCSF

Select to query event data using the OCSF Version 1.2 schema with nested object notation.

When you select the OCSF dataset, the **OCSF Category** and **OCSF Class** filters allow you to filter events by OCSF event categories and classes. See "OCSF Schema Overview" on page 39 for more information about supported OCSF classes.



3 Query builder: Optionally select different components of the AQL query. Filter conditions appear first, followed by aggr, sort, and limit.

- **Conditional statements**: Add up to ten conditions by combining the where operator with the following:
  - a field on which the conditional statement is applied.
  - an operator (`=`, `!=`, `contains`, `contains_ci`, `not contains`, `startswith`, `not startswith`, `endswith`, `>=`, `<=`, `>`, `<`, `isnull`, `isnotnull`), with logical operators `AND` and `OR`.
  - a value assigned to the field. A value is not required for the `isnull` and `isnotnull` operators.
- **Aggregation**: Apply the aggr operator to add up to three aggregation functions, including `avg`, `count`, `count_distinct`, `distinct`, `first`, `last`, `max`, `min`, `sum`, `var`. Aggregation aliases must not contain spaces.
- **Group By**: Add the Group By clause on one or more of the columns in the dataset, view, or lookup table. The Group By operator:
  - is disabled by default.
  - can be used only if aggr is also added.
  - is removed if aggr is removed.
- **Sort**: Use the sort operator to specify up to three sort fields and order them in ascending or descending order.
- **Limit**: Apply a limit to control the number of results returned, using the limit operator.
- **Output Fields**: Choose which fields to display in the results. Only selected fields appear in the final

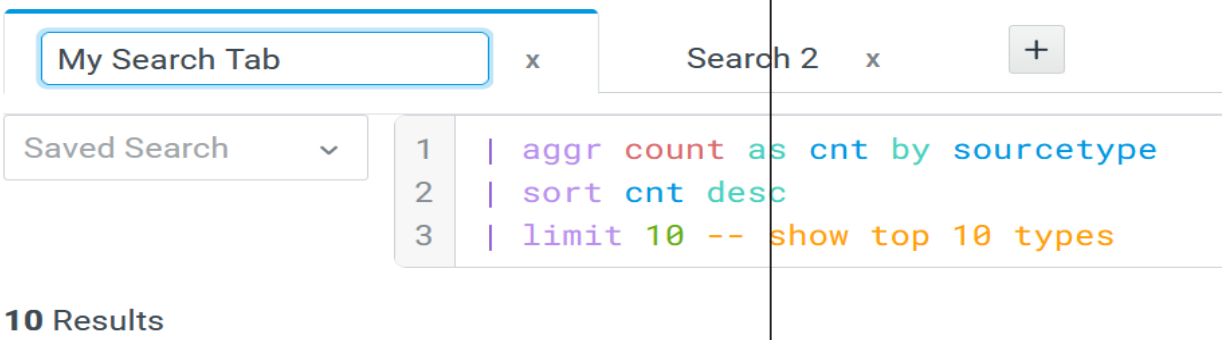
	output Additionally, click **Clear** if you would like to clear the built structured query.
4	**Time Range**: Select a relative range, quick range, or specify an absolute time range. The time period filter uses event time, which corresponds to the timestamp in the event log. **Notes**: • In the results table, event time is stored in UTC and adjusted to the web browser time zone. • The default time range specified in the **My Profile** settings does not apply to Search.
5	Search : Click **Search** to issue the search. Click the icon next to Job to stop the search. Hover the mouse over the button to view your AVC usage. See Anomali Virtual Compute for more information on how the framework measures your infrastructure usage as per your search.

6	**AQL Generator**: Use natural language to generate an AQL query. • Click **AQL Generator** then enter a natural language prompt. At this point, the **Search** button changes to **Generate AQL**. • Once you type your prompt, press **Enter** to generate the corresponding AQL query. • Click **Search** to run the query. To generate OCSF queries using AQL Generator, include the case-insensitive keyword `"ocsf"` in your natural language prompt. For example: `"Show me OCSF network activity from 192.168.1.10"` **Note:** You must include the case-insensitive keyword `"ocsf"` in your query for AQL Generator to use the OCSF schema.
7	**Saved Search**: Review a list of recent saved searches. Select a search to load the query in the search bar and issue the search. See "Saved Searches" on page 88 for more information. Search History: Review a list of recent searches; select one to load the query in the search bar and issue the search. See "Using Search History" on page 86 for more information.

8	**Focus Tabs**: Toggle between the following tabs: • **Learn AQL**: In the left panel, enter Copilot prompts to convert natural language queries into AQL expressions. See "Using Search Modes" on page 47 for more information. Alternatively, in the right panel, you can hover over the example query text and click **Run** to run one of the example AQL expressions. • **Result**: Results appear in a paged table. Review results and use the UI tools to add filters to further refine your search and set fields to be displayed in visualizations. See "Exploring Search Results and Refining Queries" on page 72 for more information. • **Visualization**: After refining a query, toggle to the **Visualization** tab to select a visualization and configure its properties, prior to add it to a dashboard. See "Using Search Modes" on page 47 for more information.

9 **Search Tabs**: Use one or more search tabs to launch a search query.

- Add up to a maximum of five search tabs.
- Switch between tabs at any time. The context of the tab is preserved even after you refresh the tab or switch between tabs.
- Double-click a tab name to rename it.
- Search using either the `eventlog` or `ocsf` schemas. See eventlog and "OCSF Schema Overview" on page 39 for more information on eventlog and OCSF field naming and conventions, respectively.



Understanding the Search Bar

Search queries are sent to the server as AQL expressions. You can enter natural language prompts or start typing a search query.

> **Note:** For non-administrator users, the OCSF filter configured in their assigned role under **Settings** > **User & Role Management** determines which OCSF data the search returns. See "Role-Based Access Control for Schemas" on page 110 for more information.

Specifying the Schema Table

Search supports querying data using two schema standards: **eventlog** and **OCSF**. Select which schema to query by specifying the table prefix before the first pipe operator. To generate OCSF queries, include the case-insensitive keyword "`ocsf`" in your natural language prompt. By default, natural language prompts generate `eventlog` queries.

Schema	Description	Syntax	Context

eventlog	Normalized Anomali schema used by default.	Specify the table name as a prefix:
		`eventlog | where src = "10.0.0.1"`
		`eventlog` is used by default:
		`| where src = "10.0.0.1"`
			Example 1
			Uses `eventlog` to run the query.
			Example 2
			No table mentioned. Prompt resolves to using default schema, which is `eventlog`.

Schema	Description	Syntax	Context

OCSF	OCSF Version 1.2 schema with support for nested-object dotted notation like `src_endpoint.ip`	Specify the table name as a prefix:
		Example 1
		`ocsf | where src_endpoint.ip = "10.0.0.1"`
		Example 2
		`ocsf | where class_name = "Network Activity" and dst_endpoint.port = 443`
			Examples 1 and 2
			`"ocsf"` keyword included. Prompt resolves to using OCSF, with `ocsf` prefix in the query.

Natural Language Searches

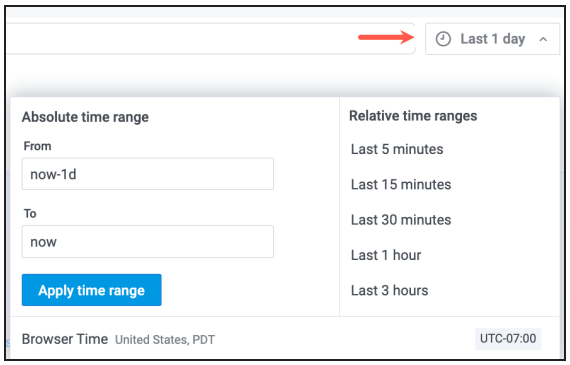
1. In the **AQL Query** Mode, enter two dashes (`--`) and then a space in the search bar. These dashes change the hint text in the search bar to "Natural Language Prompt".

2. Enter your search request in natural language. Search converts this natural language prompt to a valid AQL query. By default, natural language prompts generate eventlog queries.

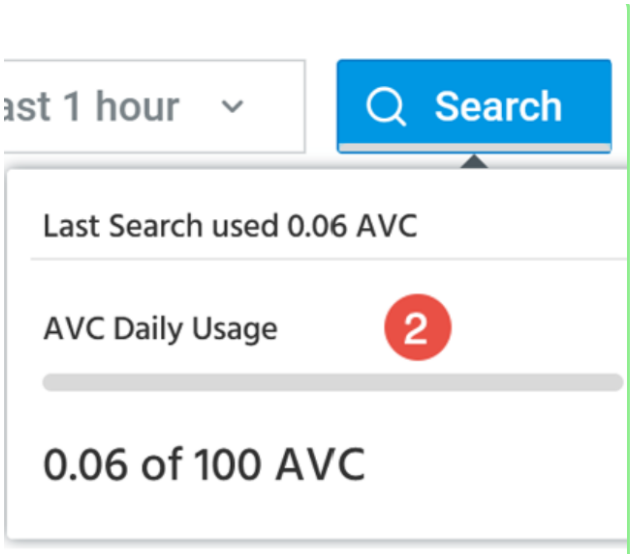
```
-- show top 10 sourcetypes
```

> **Note:** When `--` are the first characters of the first line, it invokes the natural language parser, like in Step 1. When `--` appears at the end of a line, it is treated as a comment.

3. Select a search time range from the time picker to limit your search to the selected time range.



4. Click Search. The button also displays the total AVC units you have consumed from your daily quota and the remaining balance for today. See Monitor your AVC usage for more information.



When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

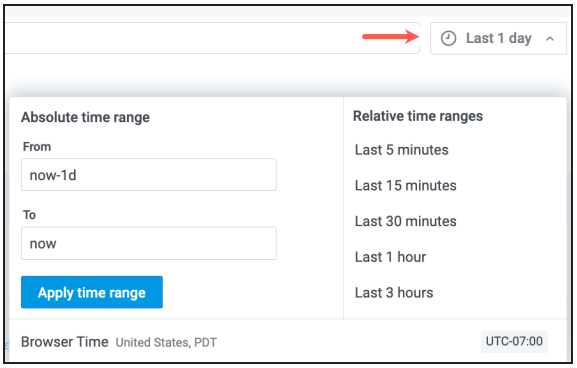
> **Note:** When `--` appears at the end of a line, it is a comment. When `--` are the first characters of the first line, it invokes the natural language parser, like in Step 1.

AQL Syntax

> **Note:** AQL boolean arguments are case-insensitive, that is, all the following boolean arguments are valid:
>
> **Positive case**: `True`, `true`, `t`
>
> **Negative case**: `False`, `false`, `f`

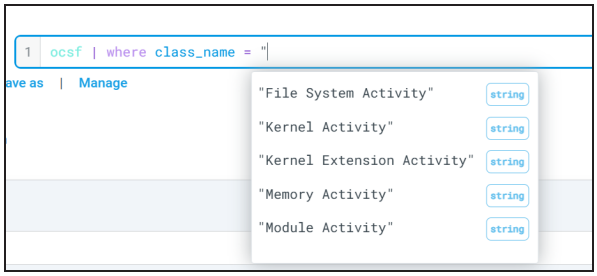
- In the Search bar, enter the operator separator `|` followed by a space.

- You can embed a subsearch in the primary search. See Subsearch for more information.
- Select a search time range from the time picker to limit your search to the selected time range.



Auto-Completion Suggestions

When you construct OCSF-based queries, you can use auto-completion suggestions that display OCSF field names with nested object paths and valid values for OCSF fields.



See "OCSF Schema Overview" on page 39 for more information on the OCSF schema, field naming conventions, and features the OCSF schema supports.

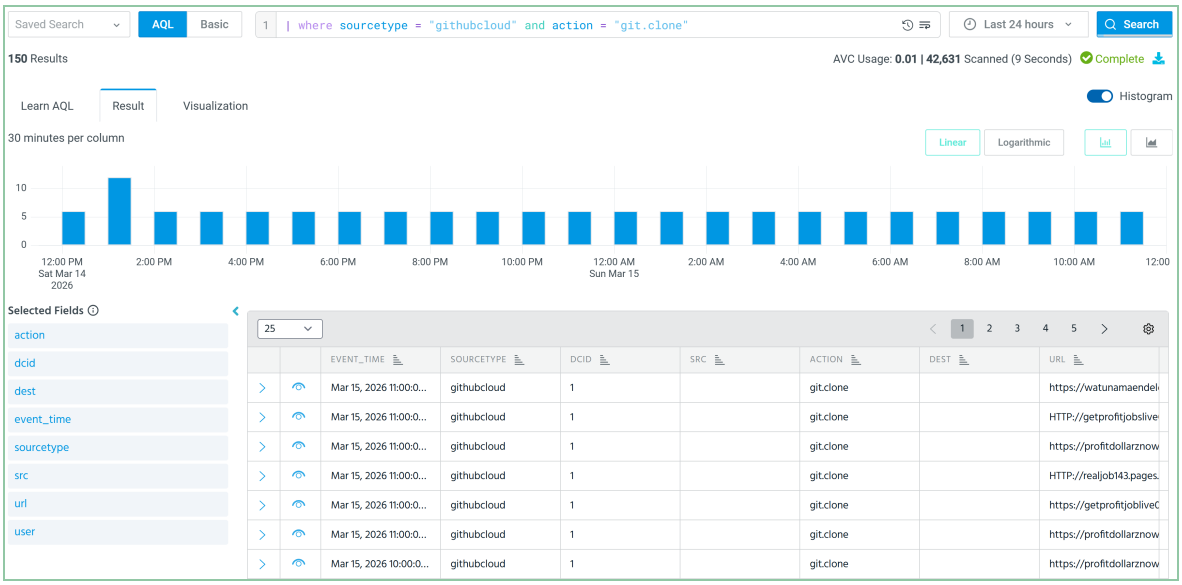
Build the `where` Clause

You can type a search expression in the search bar or use the field name selector to build a `where` clause for your search expression.

The `where` clause filters your data to selects rows that match the `where` conditions. For example, the following expression returns results that satisfy both the specified `sourcetype` and `event_name`:

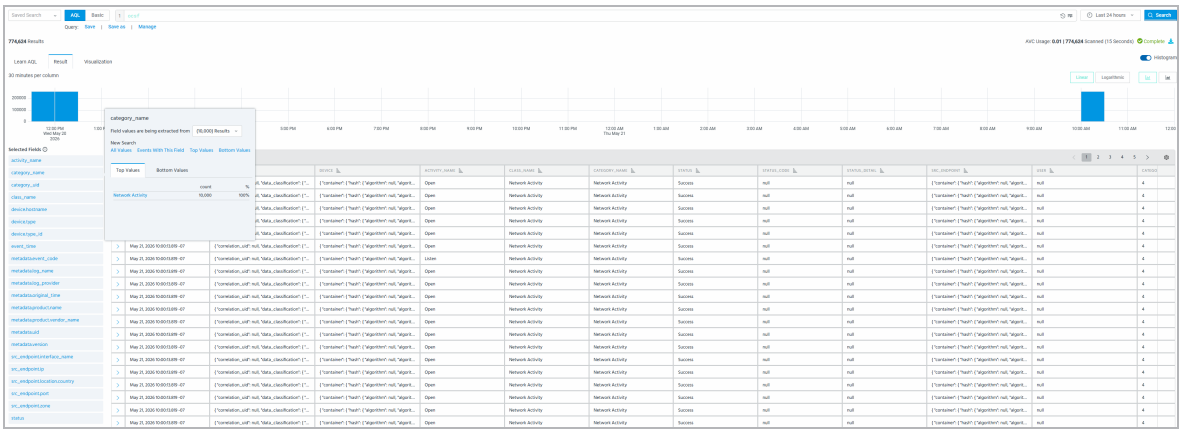
```
| where sourcetype contains "crowdstrike" and event_name contains
"LogonFailed"
```

In the example below, the query returns GitHub Cloud logs where users used the `git clone` command to copy an existing repository.



See "Using AQL Operators" on the next page to understand how you can combine the `where` clause with other operators to create advanced queries for complex use cases.

> **Note:** For non-administrator users querying OCSF data, the OCSF filter defined in their assigned role automatically constrains the rows that match the `where` clause. For example, a user with a role filter of `category_name = 'Network Activity'` sees only Network Activity events in the results, regardless of the `where` conditions in the query.



You can also embed a subsearch in the primary search to create efficent and dynamic search queries that can perform correlations across multiple datasets. See Subsearch for more information.

Using AQL Operators

AQL operators help you filter, transform, aggregate, and correlate data so you can build more precise searches and deeper investigations. For example, you can use the `timetrend` operator to perform time-based aggregations.

Use Timetrend to Discover High Frequency Events

You can use `timetrend` operator in a search expression to perform aggregations over time spans, or sort the time span aggregation rows by count. Additionally, use the `span` argument to define the time span in seconds, minutes, hours, days, weeks, months, or years.

In the following example, the `timetrend` operator counts instances of events with the same user over one day time spans.

Notes:

- You cannot use the `fields` operator with `timetrend`. However, you can use `by` to list the fields to include in the results table. For example, the query `timetrend count by url, src span = 2d` includes the fields `url` and `src` as columns in the results table.

```
timetrend count by url, src span = 2d
```

- After you issue a search with `timetrend`, the value for `span` may be recalculated and rationalized to create meaningful buckets, based on the configured time range.

Use Aggregation to Analyze OCSF Network Activity

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

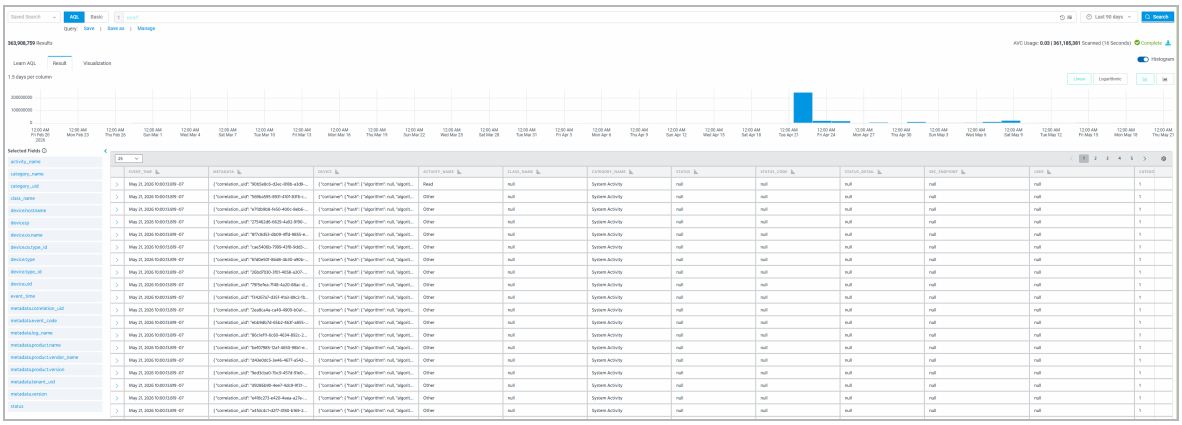
You can use the `aggr` operator with the OCSF Version 1.2 schema to perform aggregations on nested fields, ensuring the same analytical capabilities as `eventlog` queries while using the OCSF schema structure.

In the following example, the query counts OCSF network activity events grouped by source endpoint IP address and destination port.

```
ocsf | where class_name = "Network Activity" | aggr count as cnt
by src_endpoint.ip, dst_endpoint.port | sort cnt desc
```

> **Note:** For non-administrator users querying OCSF data, the OCSF filter defined in their assigned role constrains the data that AQL operators process. Aggregations, sorts, and other operators run only against the OCSF categories the user has permission to access. In the example below, a user whose role permits both **Network Activity** and **System Activity** runs `ocsf | aggr count as cnt by category_name` and sees exactly those two categories in the results — no other categories are returned regardless of what is in the data.

The **Max Search Back** setting in the role also constrains OCSF queries. Even if a user selects a wider time range from the time picker, results are returned only for the period permitted by the Max Search Back value in their role. In the example below, the user selected **Last 90 days** from the time picker, but their role sets Max Search Back to 30 days — the histogram shows data only for the last 30 days, while the x-axis extends further back with no results outside that window.



Linking from External Platforms

When you drilldown into Anomali Search from an external platform (for example, lookups from a SIEM), you can use the following URL parameters:

Case 1: Only Search query with time range for new searches

To start a new search, for an existing query, you can include only the query in the URL along with the time ranges:-

Absolute time format

```
xdr/gf/m/a?query=query&from=1740521149989&to=1740521449989
```

Relative time format

```
xdr/gf/m/a?query=query&from=now-3d&to=now
```

Case 2: Only Job ID for existing, completed searches

For an existing search that is already complete, you can provide only the `job_id` the URL to load and view the corresponding search results.

```
xdr/gf/m/a?job_id=job_id
```

Search with Anomali Copilot

On the Search page, you can use Anomali Copilot to run AQL queries generated from either natural language prompts or pre-defined exploratory suggestions.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Searching with AQL Generator

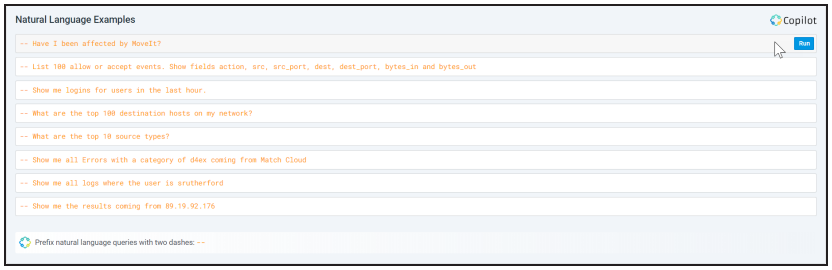
You can use the **AQL Generator** to run a search query constructed from a natural language prompt, as follows:

1. Click **AQL Generator**.

2. Enter your own question in the **Search** bar in natural language.
3. Click **Generate AQL**. This generates the search query for the prompt you entered.

4. Once the query is generated, click **Search** to launch a search.

Alternatively, hover over one of the **Natural Language Examples** and click **Run** to add the question to the **Search** box. Clicking **Run** automatically executes Steps 1–4 mentioned above.



Searching Without Using AQL Generator

You can also launch an AQL search using double dashes (--) followed by your natural language search prompt.

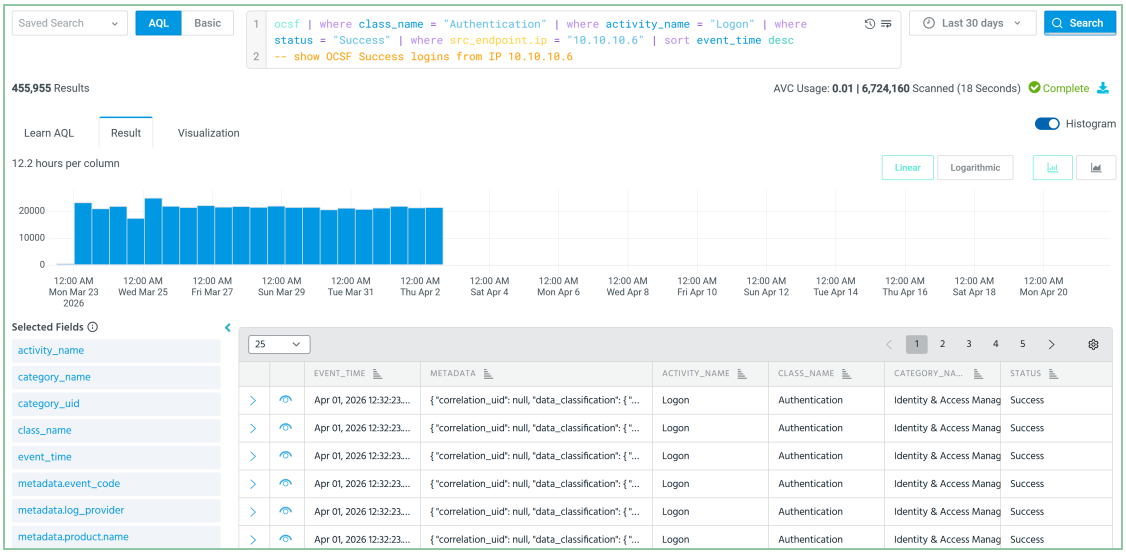
To search directly use natural language prompts:

1. In the Search bar, enter double dashes (--) and then enter your natural language prompt.

2. Click **Search**. This initiates the process of converting your natural language prompt into an AQL query.

3. Once the conversion is complete, the Search bar now displays the corresponding AQL query.

4. Click **Search** to complete the search and obtain the search results.



Supported OCSF Classes for Anomali Copilot

Anomali Copilot supports the following OCSF classes:

OCSF Category	Description
API Activity	API call events capturing request and response details, including REST endpoints, methods, status codes, and resource identifiers for cloud and application service interactions.
Account Change	User account lifecycle events including account creation, deletion, enabling, disabling, password changes, and attribute modifications.
Authentication	Authentication and session events including login attempts, logon successes and failures, multi-factor authentication, and single sign-on activities.
Email Activity	Email handling events covering message sending, receiving, delivery, forwarding, and filtering activities, including attachment and email metadata.
Entity Management	Lifecycle events for managed entities such as configurations, policies, certificates, and organizational objects, including creation, modification, and deletion.
Filesystem Activity	File system operation events including file creation, deletion, modification, access, and attribute changes across local and network storage.
Group Management	Group lifecycle events covering group creation, deletion, and modification, including membership additions and removals.
HTTP Activity	Web traffic events capturing HTTP and HTTPS request and response details, including URLs, methods, status codes, user agents, and payload metadata.
Network Activity	Network connection and traffic events including TCP/UDP sessions, flows, firewall logs, and network protocol activities between endpoints.
Process Activity	Process lifecycle events including process launch, termination, injection, and other activities that describe process execution and behavior on a host.

Natural Language Queries with OCSF

The **AQL Generator** supports generating queries for both eventlog and OCSF schemas. By default, natural language prompts generate eventlog queries. To generate OCSF queries, include the case-insensitive keyword "`ocsf`" in your natural language prompt.

Examples:

Natural Language Prompt	Generated Query Type	Reason
`Show me network traffic from 192.168.1.10`	`eventlog` query	No table mentioned. Prompt resolves to using default schema, which is `eventlog`.
`Show me OCSF network activity from 192.168.1.10`	OCSF query with `ocsf` prefix as follows: `ocsf | ...`	OCSF keyword included. Prompt resolves to query with `ocsf` prefix for failed authentication events.
`Query OCSF for failed authentication events`	OCSF query that filters the Authentication OCSF class	OCSF keyword included. Prompt resolves to query with `ocsf` prefix for filtering the Authentication OCSF class.
`Find ocsf events where destination port is 443`	OCSF query that uses the `dst_endpoint.port` OCSF field using dotted notation.	OCSF keyword included. Prompt resolves to query with `ocsf` prefix to use the specified destination port.

> **Note:** When generating OCSF queries, the **AQL Generator** uses OCSF field naming conventions such as `src_endpoint.ip`, `dst_endpoint.port`, or `actor.user.name`. See "OCSF Schema Overview" on page 39 for more information about OCSF field naming conventions.

See the Anomali Copilot documentation to learn more about the Anomali Copilot suite of generative AI solutions that supports AI-enabled detections, investigations, and incident response management.

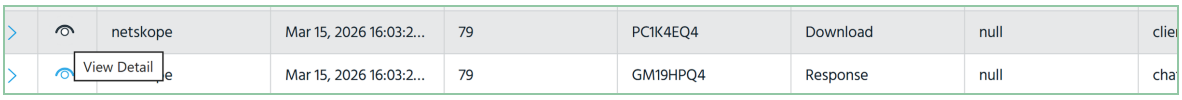
Exploring Search Results and Refining Queries

The Search interface provides multiple options to gain deeper insight into your search results and refine your search query.

Exploring Search Results

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

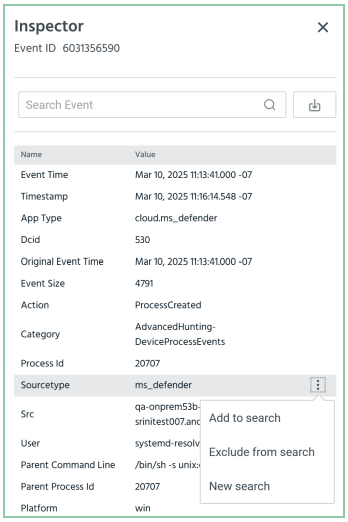
A typical Search — whether based on the eventlog or OCSF Version 1.2 schema — produces results that you can inspect as follows:



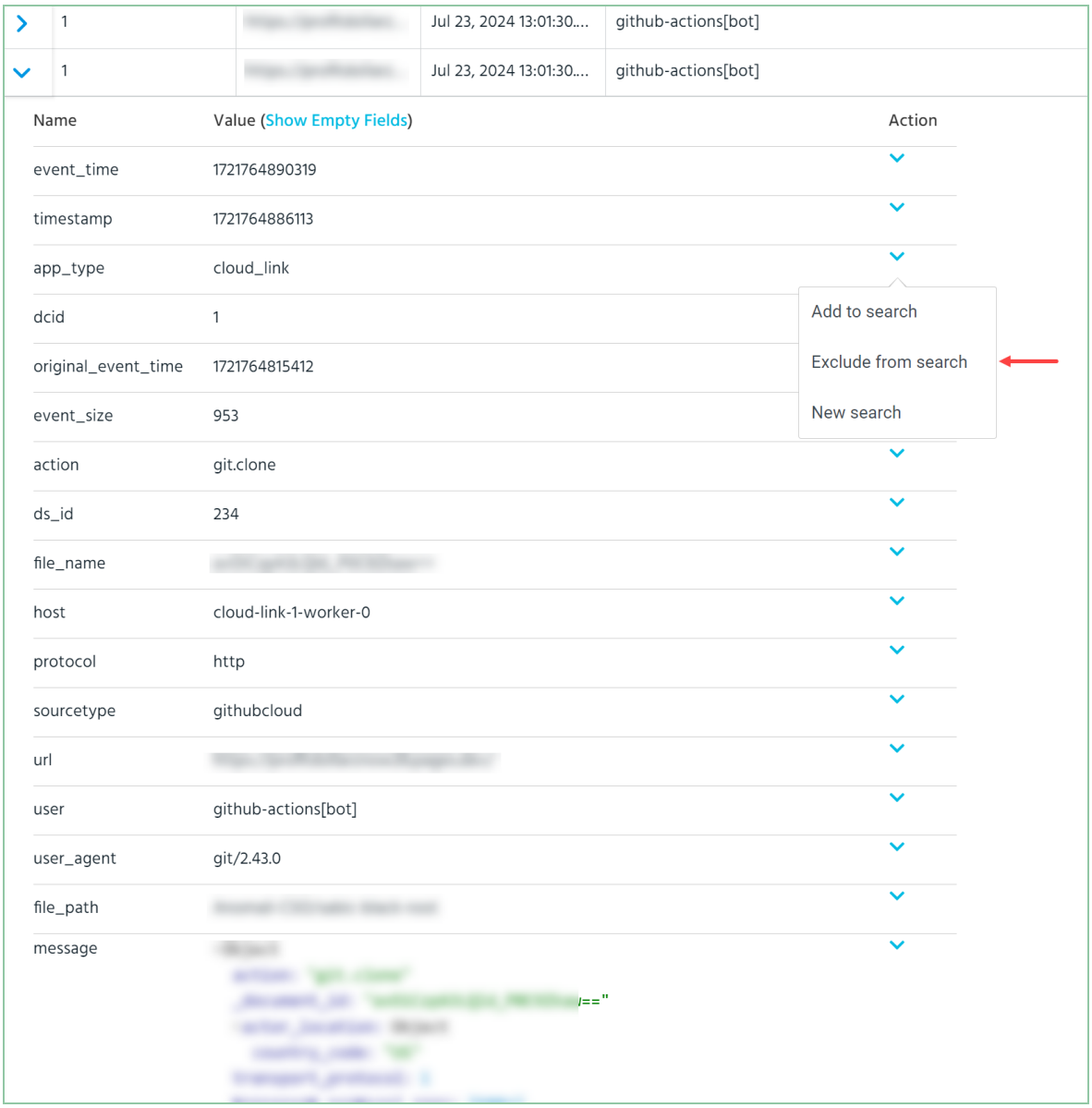
- Hover your mouse on the search result and click the **View Detail** ( ) icon to open the **Event Inspector** side panel and view the details about that particular result row.

  In the example screenshot above, the **Event Inspector** side panel displays the **Event ID**, the different fields related to the search result, and the **Search Event** field that lets you search for a particular field by text. You can also add a field to your search query, exclude a field from your search query, or launch a new search with a field. Click the **Download CSV** icon ( ) to download the result as

a CSV file.



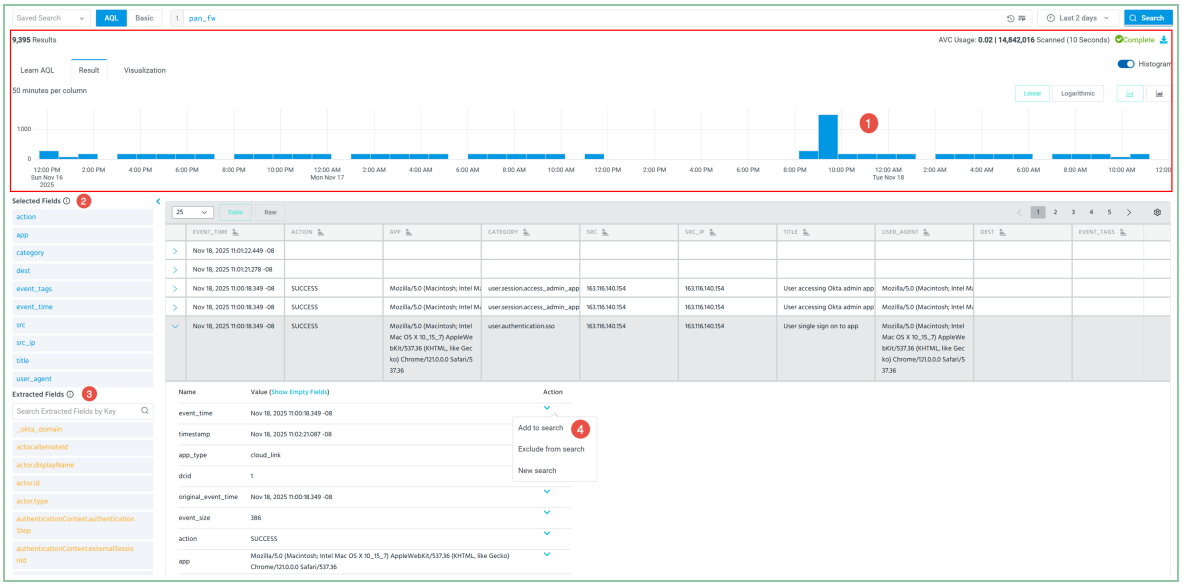
- Click the > icon on the left side of the row to expand its details. In the expanded row, click the ⌄ icon on the right side to display the **Field Values** filter menu. The following example shows the **Field Values** filter used to exclude results for a specific `app_type` field value. As a result, the field-value pair to exclude is added to the `where` clause in the search expression. Click Search to issue the new search with the field excluded.



> **Note:** The user interface elements for refining queries are not available after you submit a query using the `aggr` or `timetrend` operators because the results for those queries are aggregations.

Refining Search Expressions

The **Result** tab displays Search results as a paged table. Review results and add filters to further refine your search and set fields to display in a search visualization in the **Visualization** tab.



1	**Time Series**: The graph plots results over time. You can display the time series graph as a bar graph, a line graph, or points. If you click and drag over the graph elements (bars for example), the time range selector updates to the selected range and accordingly issues a new search.

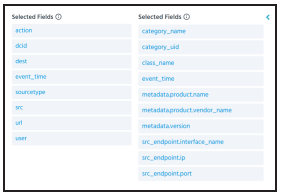
2

Selected Fields: These fields are schema fields that are currently chosen for display in the results grid. As shown in the example screenshot below, depending on your query, these could be eventlog fields or OCSF Version 1.2 schema fields.

OCSF Field Extraction

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

- When querying OCSF Version 1.2 fields, nested object fields are displayed using dotted notation, for example, `src_endpoint.ip`, `metadata.product.vendor_name`, or `src_endpoint.port`.
- These OCSF fields can be used in drill-down actions, added to search expressions, or excluded from results, just like eventlog fields.
- The drill-down options allow you to filter, aggregate, or create new searches based on OCSF nested field values. See "OCSF Schema Overview" on page 39 for more information about OCSF Version 1.2 field naming conventions and structure.



When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

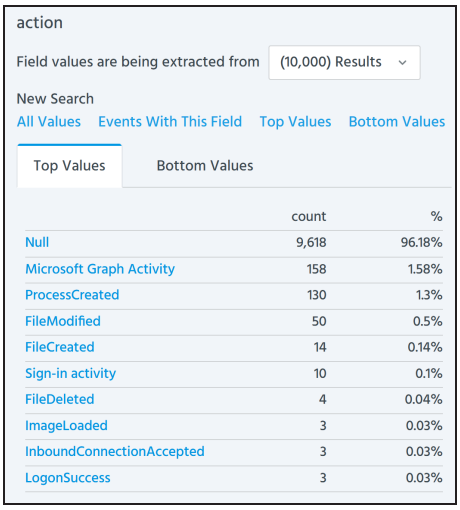
Using the Pane Statistics

The **Selected Fields** pane displays statistics about the event

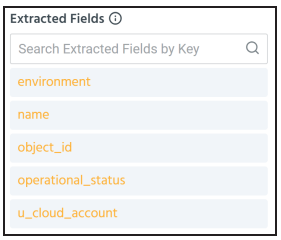
	fields, including: • **Overview**: Provides a quick overview of the data fields and their characteristics for the current search results. • **Field Insights**: Identifies key fields for analysis. • **Anomaly Detection**: Spots anomalies and unexpected values that might indicate security incidents. • **Efficient Filtering**: Enables filtering and drilling down on current search results into specific data subsets for more detailed investigation. • **Data Quality**: Assesses the quality and completeness of the data to ensure critical fields are populated correctly. On the pane: • Field values are extracted from the top 10,000 results. Select **All Results** to view statistics for all the fields in the result set. Note that this action will consume additional AVC units. See Anomali Virtual Compute for more information on how AVC usage works. • Click the field name to display the field statistics. Use the **Top Values** and **Bottom Values** tabs to view the top 10 and bottom 10 unique values for the field as well as their count and percentage in the search results. • Click the field in the **Top Values** or **Bottom Values** tabs to add that field-value pair to the search expression `where` clause. The `where` clause filters search results. • You can also launch a new search by clicking any of the following search suggestions: ▪ **All Values**: Launches a new search that calculates the aggregated count of the selected field in the pane. In the following screenshot, clicking **All Values** calculates the aggregated count for the `action` field. ▪ **Events With This Field**: Issues a new search that returns results wherein the selected field in the pane is not null. In

the following screenshot, a new search returns results that contain the `action` field.

- **Top Values**: Launches a new search that returns the top 10 results for the selected field in the pane. In the following screenshot, the new search will return the top 10 `action` fields.
- **Bottom Values**: Launches a new search that returns the bottom 10 results for the selected field in the pane. In the following screenshot, the new search will return the bottom 10 `action` fields.



	count	%
Null	9,618	96.18%
Microsoft Graph Activity	158	1.58%
ProcessCreated	130	1.3%
FileModified	50	0.5%
FileCreated	14	0.14%
Sign-in activity	10	0.1%
FileDeleted	4	0.04%
ImageLoaded	3	0.03%
InboundConnectionAccepted	3	0.03%
LogonSuccess	3	0.03%

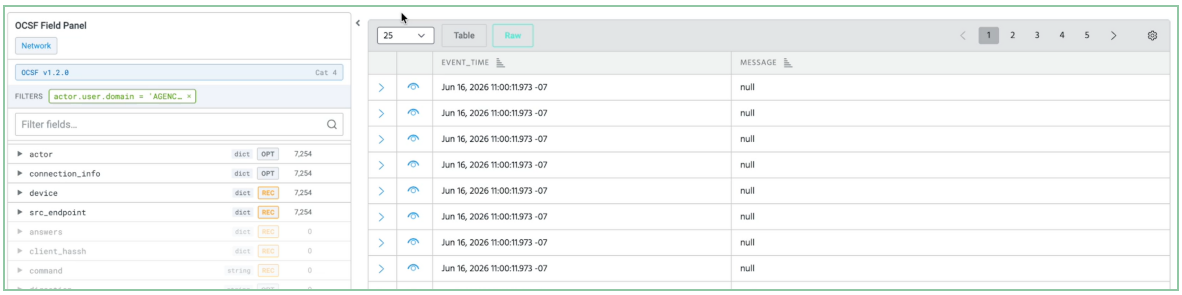
3	**Extracted Fields**: Contains all the fields that are automatically extracted from the raw message (JSON, CEF, or key-value pairs). The benefit of such on-the-fly extractions is that it enables analysts to rapidly and flexibly view and use these extracted fields in AQL queries to conduct downstream analysis across different Anomali applications, even if they are not normalized into eventlog schema fields. You can also search for an extracted field by key and analyze an extensive list of fields extracted directly from the raw message. See Dotted Notation to understand how you can use these extracted fields in your searches. 
4	**Event Field Values**: Click the > icon on the left side of the results table to expand a result row and view event details. In an expanded row, click the ⌄ icon on the right side to display a Field Value filter menu with the following options. • **Add to search**: Adds the field-value pair to the search `where` clause. The `where` clause filters search results. • **Exclude from search**: Adds the field-value pair to the search `where` clause with logic to exclude results matching the field-value pair. • **New search**: Clears the current search expression and builds a new search expression starting with a `where` clause with this name-value pair.

OCSF Field Panel

The OCSF Field Panel is a collapsible sidebar on the left pane of the Event Search page. When you run an OCSF search, the panel lists the schema fields for the active OCSF category, shows how many events in your current result set have each field populated, and lets you add include or exclude filters to your active query by clicking, without typing field names manually in the search bar.

Notes:

- The OCSF Field Panel is available only when you are searching OCSF data. Prefix your query with `ocsf` to search OCSF events. See "OCSF Schema Overview" on page 39 for more information.
- Field counts are calculated before transformations such as dedup, calculated fields, or joins, therefore, the numbers may be higher than expected.
- In specific situations such as a heavy system load, only a subset of fields may temporarily appear. This issue can be rectified with subsequent searches.
- Fields in the Field Panel only applies to standard, non-aggregated, non-pivoted OCSF searches.



OCSF Field Panel

The OCSF Field Panel contains the following elements.

- **Category tab**: Indicates the active OCSF Version 1.2 category.
- **OCSF v1.2.0 / Cat N**: The schema version and the numeric identifier of the active category.

- **Filters**: Displays the filter conditions currently active in your query.
- **Filter fields**: Type to narrow the field list by field name.
- **Field list**: Lists the OCSF schema fields for the active category, along with their data type, requirement level, and hit count.

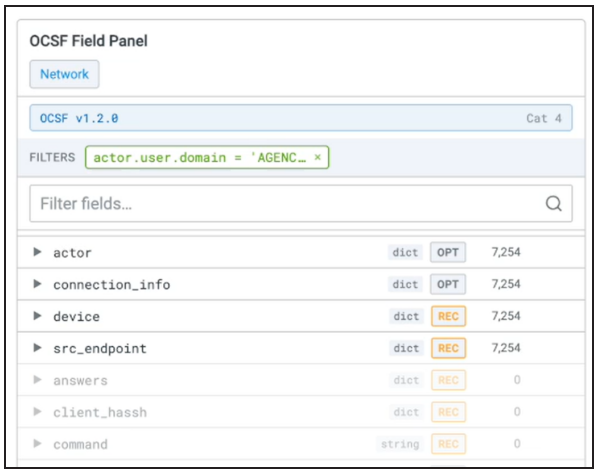
Category Tabs

The **OCSF Field Panel** groups fields by the six top-level OCSF Version 1.2.0 categories.

Tab Label	Category Number	OCSF Category
Sys	1	System Activity
Findings	2	Findings
IAM	3	Identity & Access Management
Network	4	Network Activity
Discovery	5	Discovery
App	6	Application Activity

The **OCSF Field Panel** appears only when your active query filters on exactly one OCSF category using `category_name`. The panel displays the fields for that category and the tab for the matching category is active. If your query specifies no category filter, or more than one category, the search runs as a standard OCSF search and the **OCSF Field Panel** is not displayed.

Field List



Each row in the field list displays the following information:

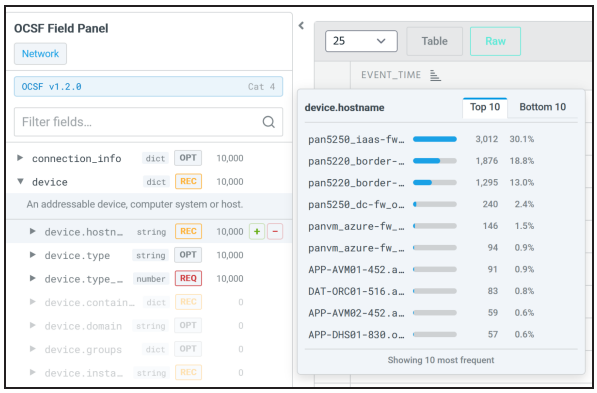
- Click ▶to expand the field and view its description. Click ▼ to collapse.
- The field name in dotted notation, for example, `src_endpoint.ip` or `actor.user.name`.
- The field data type, for example, `dict`, `string`, or `number`.
- The requirement badge: **REQ** (Required), **REC** (Recommended), or **OPT** (Optional).
- The hit count: the number of events in the current result set that have this field populated.

Fields with no hit count are dimmed but remain visible, so you can discover all available schema fields for the active category.

Expanding a Field

Click ▶next to a field name to expand the row and view the field description from the OCSF schema. Click ▼ to collapse the row. Expanding a field does not trigger any additional data requests.

Field Value Distribution



Hover over a field row to open the field value distribution. The panel has two tabs:

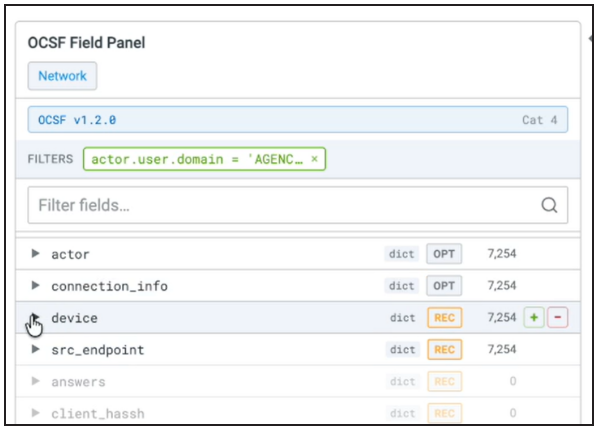
- **Top 10** (default): The 10 most frequently occurring values for this field in the current result set, ranked in descending order by event count.
- **Bottom 10**: The 10 least frequently occurring values for this field in the current result set, ranked in ascending order by event count.

Each value row displays the value, a proportional bar, the event count, and the percentage of the result set. Click **+** or **−** on a value row to add an include or exclude filter for that specific value.

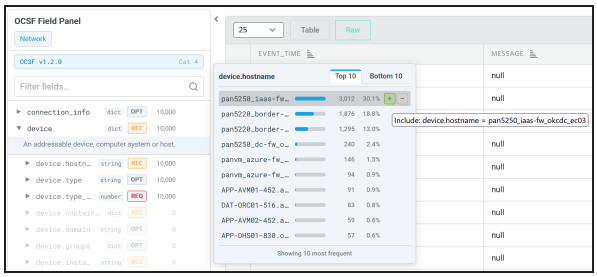
Filtering the Field List

Type in the **Filter fields** box at the top of the panel to show only the field rows whose names contain the text you enter. Clear the box to restore the full field list for the active category.

Click-to-Filter



Hover over a field row to reveal the **+** (include) and **−** (exclude) action icons. Clicking either icon appends a filter clause to your active query and re-executes the search automatically. The following table describes the clause generated for each action:



Action	Generated Query Clause	What it means	Example
Click **+** on a field	`AND field_name:*`	Show only events where this field has any value populated.	`AND src_endpoint.ip:*`
Click **+** on a value	`AND field_name:"value"`	Show only events where this field equals this specific value.	`AND src_endpoint.ip:"192.0.2.1"`
Click **−** on a field	`AND NOT field_name:*`	Exclude all events where this field has any value populated.	`AND NOT src_endpoint.ip:*`
Click **−** on a value	`AND NOT field_name:"value"`	Exclude all events where this field equals this specific value.	`AND NOT src_endpoint.ip:"192.0.2.1"`

If the query bar is empty when you apply a filter, the leading `AND` is omitted. The applied filter also appears in the query bar so you can view and edit the full query. To remove an applied filter, click its active icon again, which removes the clause and re-executes the search. When a filter is active, the field row **+** icon turns green for an include filter and the **−** icon turns red for an exclude filter, so you can see at a glance which fields have filters applied.

Building the Fields List

Note:

- Field names and aliases are case-insensitive. Field values and string literals are case-sensitive.
- For example, both the following example queries return the same results.

Query 1: `|where sourcetype != "" |calc Date = strftime(event_time, '%Y-%m-%d')`

Query 2: `|where sourcetype != "" |calc date = strftime(event_time, '%Y-%m-%d')`

To facilitate hunting, you can customize the columns displayed in the results table and specify the sort order. The `fields` list determines the columns that appear in the results table. You can type a `fields` list in the search bar or use the **Field Name** selector to build it. The `sort` clause specifies the field to sort on and the sort order (ascending or descending).

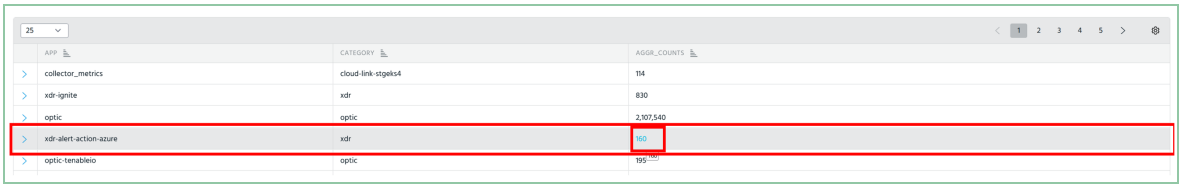
You can also embed a subsearch in the primary search to create efficent and dynamic search queries that can perform correlations across multiple datasets. See Subsearch for more information.

Aggregating, Sorting, and Limiting Result Rows

You can use the `aggr` operator in a search query to perform counts and sort the aggregated rows by count, gaining insight into the comparative prevalence of field values, as shown in the example below. The `sort` and `limit` operators are used to further refine the result set.

From the aggregated results, you can click a hyperlinked value to issue a new search that has the selected value as the `where` operator filter.

You can also drill down on aggregations performed on multiple fields.



APP	CATEGORY	AGGR_COUNTS
collector_metrics	cloud-link-stgeks4	114
xdr-ignite	xdr	830
optic	optic	2,107,540
xdr-alert-action-azure	xdr	160
optic-tenableio	optic	195

In the example above, clicking the hyperlinked aggregated value `160` in the `Aggr_Counts` column drills down into that specific aggregation and issues a new search with `app=xdr-alert-action-azure` and `category=xdr`.

You can also embed a subsearch in the primary search to create efficent and dynamic search queries that can perform correlations across multiple datasets. See Subsearch for more information.

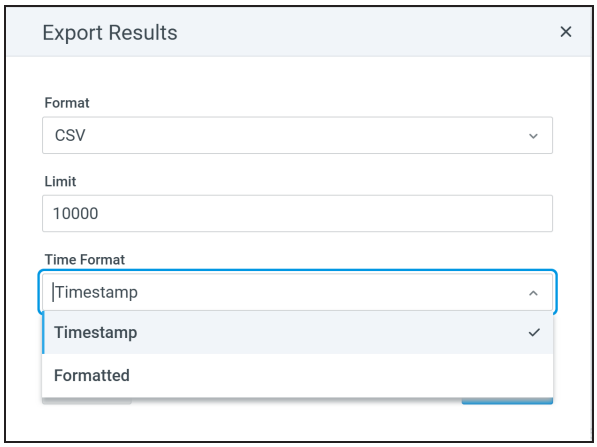
Using Search History

Search saves your 10 most recent searches.

To issue a search from your search history:

1. Click the **Search history** icon ( ) to display the list.
2. Select a query to run the search with the previous AQL syntax and absolute time range.

Exporting Search Results



To export event search results:

1. On the Search results page, click **Export** ( ).
2. In the **Export Results** dialog box:
   a. Select CSV or JSON as the export format.
   b. Enter the maximum number of search results you want to export. The default limit is 10,000.
   c. Select the date and time (in UTC) format for the exported results, as captured in the `event_time` field:

i. **Timestamp**—Export date and time values as UNIX timestamps. For example, `1776465184000` is the UNIX timestamp for the exported result.

ii. **Formatted**—Export date and time values in a human-readable date-time format. For example, `2026-04-17 10:37:19 PM` is the equivalent human-readable time format for the exported result.

3. Click **Submit** to initiate the export.

Guidelines for Search

Anomali recommends the following guidelines to optimize your Search experience.

Search Throttling

If the total number of bytes scanned across all your search queries exceeds 50 TB from the start of a given day (in UTC), your search time range for any additional queries will be limited to a 24-hour window until the end of that day (in UTC). This limit will reset the following day.

For example, if by Monday, 1:00 PM UTC, 12 of your search queries add up to 50 TB of scanned data, then your search time range will be limited to a 24-hour window till Tuesday 12:00 AM UTC, after which the limit resets.

Users, Roles, and Role-based Search

- By default, every search can span data that is 90 days old.
- Organization Administration can configure the **Max search time range** parameter to limit the search range, while creating or updating a role. See User and Role Management on the Anomali Platform for more information
- If you are assigned more than one role, the highest search range will apply. For example, if you are assigned role A that is configured with a search range of 6 months and role B that has a two-year range, then you will be able to search up to two years back in time.
- You will encounter an error if you select a search range higher than the range configured for your role. In such a case, you will either have to:
  - Reduce your search time range.
  - Use Turbosearch.

In the example above, if your org administrator has configured 10 days as the maximum search range for your role, you will encounter an error if you try to search event logs in the last 11 days.

Recommended actions

- Select a search range that is lesser than or equal to the range assigned to your role by your organization administrator.
- Contact your organization administrator to be assigned a different role that supports a higher search range.
- Use Turbosearch if you necessarily need a search range higher than the range assigned to your role by your organization administrator and if you cannot be assigned a different role with higher search range.

See User and Role Management on the Anomali Platform for more information on how role-based access works on the Anomali platform.

In addition to the eventlog access controls described above, organization administrators can define OCSF-specific data filters in user roles to restrict access to particular OCSF categories. See "Role-Based Access Control for Schemas" on page 110 for more information.

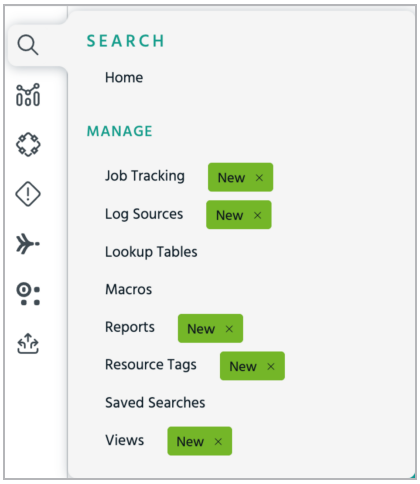
Saved Searches

Saved searches let you store frequently used AQL queries. You can also apply tags to locate and organize saved searches, as well as filter, share, and edit them directly from the Saved Searches list.

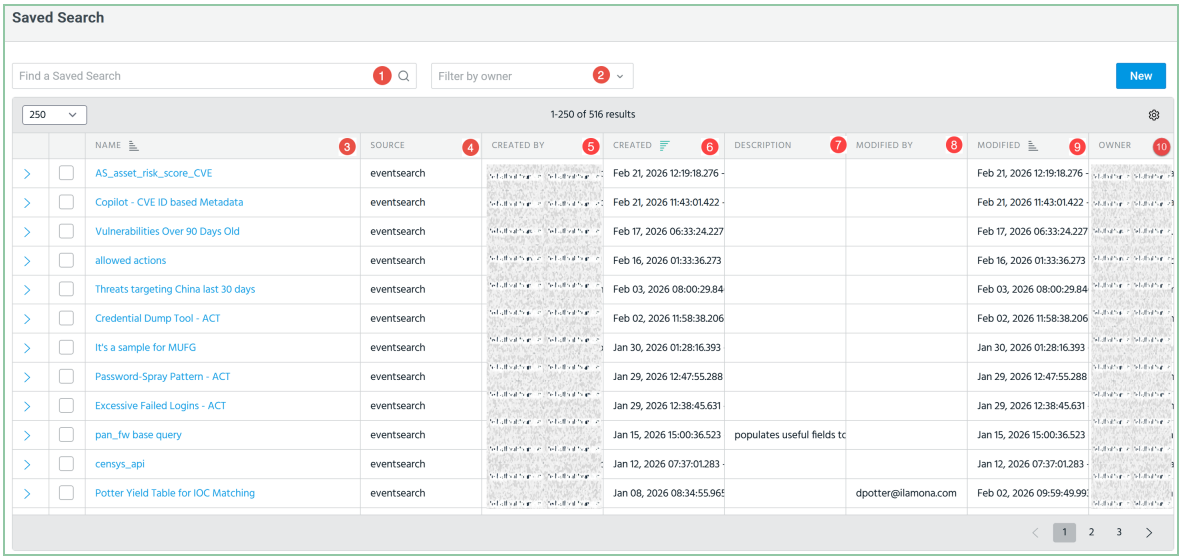
When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Accessing Saved Searches

To access all saved searches, on the left navigation menu, click **Search** > **Manage** > **Saved Searches**.



Understanding the Saved Searches Catalog



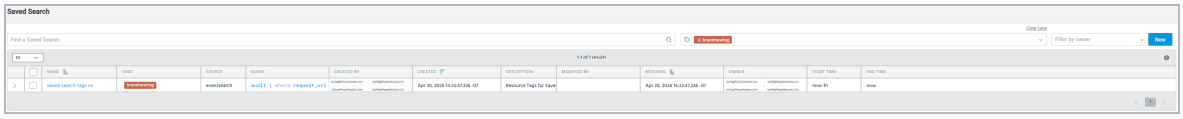
1	**Find a Saved Search:** Enter the name of the saved search you want to find. The search is incremental, that is, the results are updated as you type.

	Filter by tag: Filter the saved search list by one or more resource tags. • The drop-down shows only tags you own or that are shared to your organization. Organization administrators can also see private tags owned by other users. • Selecting multiple tags returns only saved searches that have all selected tags applied. • Click **Clear tags** to remove the filter. • The count next to each tag reflects usage for saved searches only, not the total across all the supported resource types. See Filtering saved searches by tag for details.
2	**Filter by owner**: You can filter saved searches by owner. Possible values include: • **Mine**: returns saved searches for which you are the owner. • **System**: returns out-of-the-box saved searches, that is, saved searches that are owned by nobody and that cannot be changed. • **Not System**: returns saved searches that are not owned by system, such as saved searches created by a user. • **<user>**: returns saved searches where the selected user is the owner.
3	**Name:** Name of the saved search.
	Tags: Resource tags applied to the saved search, displayed as colored chips. To filter the list by tag, see "Filtering Saved Searches by Tags" on the next page.
4	**Source:** The source of the saved search.
5	**Query:** The AQL query associated with the saved search.

6	**Created by**: Indicates the creator of the saved search. By default, the creating user is the owner and is private (not shared). The owner can: • Modify all the details of the saved search. • Configure sharing or permissions granted to users. • Delete the saved search.
7	**Created:** The date and time when the saved search was initially created.
8	**Description:** A brief explanation of the purpose or functionality of the saved search.
9	**Modified by:** The user who last updated the saved search.
10	**Modified:** The date and time when the saved search was last updated.
11	**Owner**: The user who owns the saved search.
12	**Start Time**: The start time of the search duration range associated with the saved search. This time can either be relative or absolute time.
13	**End Time**: The end time of the search duration range associated with the saved search. This time can either be relative or absolute time.
14	**New:** Click **New** to create a new saved search.

Filtering Saved Searches by Tags

Use the tag filter on the Saved Search list to show only saved searches that have specific resource tags applied.



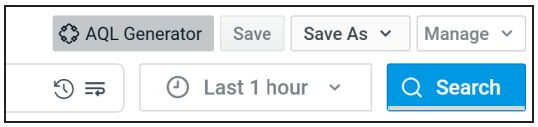
> **Note:** Organization administrators can see all tags, including private tags owned by other users.

To filter saved searches by tag:

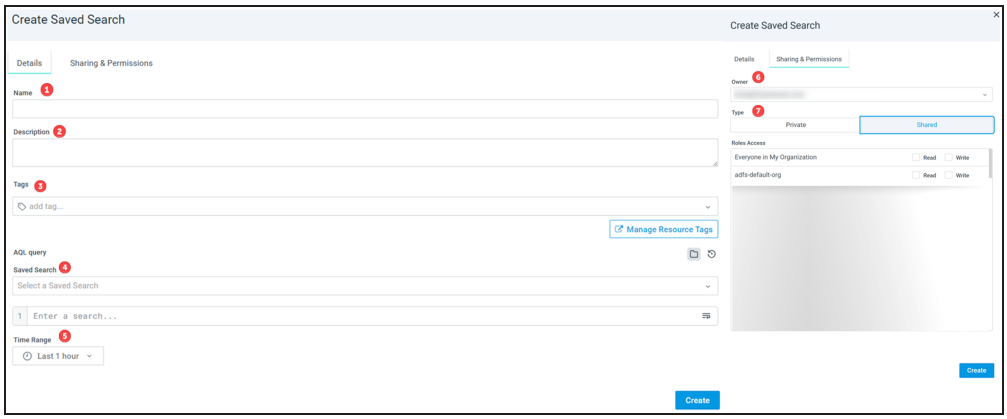
1. Navigate to **Search** > **Manage** > **Saved Searches**.
2. Click **Filter by tag** and select one or more tags from the dropdown. After selecting these tags:
   - The list updates to show only saved searches that have all selected tags applied. For example, selecting the tag **brandnewtag** returns only saved searches containing this tag.
   - The dropdown shows only tags you own or that are shared to your organization.
   - The count displayed next to each tag reflects usage for saved searches only, not the total across all the supported resource types. A tag used across multiple resource types will show a different count on each resource page, depending on the resource you are viewing.
3. Click **Clear tags** to remove the filter and restore the full list.

Saving a Search

To save a search, click **Save As** on the top-right. If you edit an existing saved search, click **Save** to save the changes you made.



Creating a Saved Search

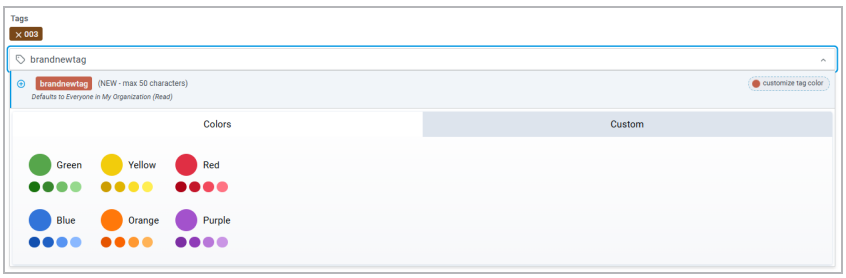


To create a saved search, navigate to **Search** > **Manage** > **Saved Searches**.

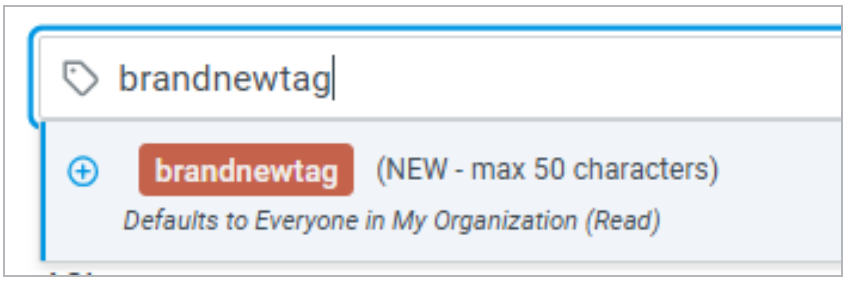
1	**Name**: Enter a name for the search query you want to save.
2	**Description**: Enter a description for the search.

3 **Tags**: Apply one or more resource tags to the saved search.

- The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.
- To apply an existing tag, type in the field and select it from the auto-complete suggestion results.
- To create a new tag, type a name that does not match any existing tag and confirm the selection. A color picker appears so you can choose a tag color before the tag is created.



- New tags default to **Everyone in My Organization (Read)**. You can change this later from the "Resource Tags" on page 146 settings page.
- A new tag is added to the shared tag list immediately, even if you abandon the edit without saving the saved search.



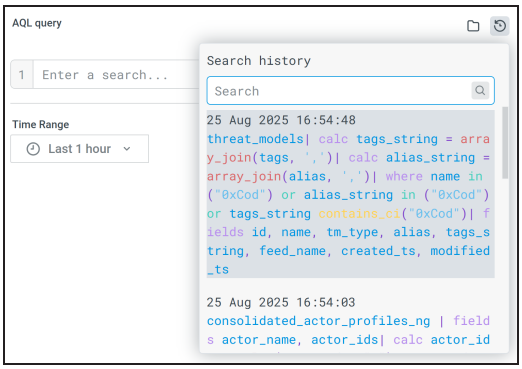
- Applied tags render as colored chips. Click the cross icon (×) on the tag chip to remove the tag from the saved search. This does not delete the tag from the catalog. Any user with edit access to the saved search and visibility to a tag can remove the tag.

> **Note:** The tag-to-saved-search association is saved only when you save the saved search. Abandoning the edit does not apply the tag, even if a new tag was created in the catalog during that session.

Click **Manage Resource Tags** to open the "Resource Tags" on page 146 settings page, where you can view and manage all tags you own or can access. Organization administrators can see all tags.

4 **AQL Query**: You can supply an AQL query for a saved search in the three following ways:

1. Select a saved search from the list of saved searches. This list only displays saved searches for which you have permissions.
2. Enter a valid AQL Search query that you would like to save. You can build queries using both eventlog and OCSF schemas. See "OCSF Schema Overview" on page 39 for more information on OCSF field naming and conventions.
3. Click the history icon ( ) on the right to search the history of your saved searches and select an AQL query. This list only displays saved searches for which you have permissions.

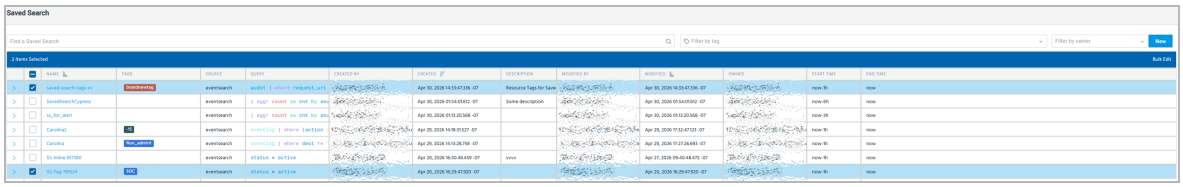


5 **Time Range**: Select a time range from the time picker. You can select either an absolute or relative time range.

6	**Owner**: The owner of the saved search. By default, this is the user who created it. Owners can modify all details, configure sharing and permissions, and delete the saved search. You can transfer ownership to another user.
7	**Type**: Set the visibility of the saved search: • **Private**: Only the owner can see and use the saved search. • **Shared**: The saved search is accessible based on the **Roles Access** settings. By default, newly created saved searches are shared with **Read**-only access for everyone in the organization. Users with **Write**-only permissions can read or modify the saved search but cannot delete it.

Bulk Editing Tags on Saved Searches

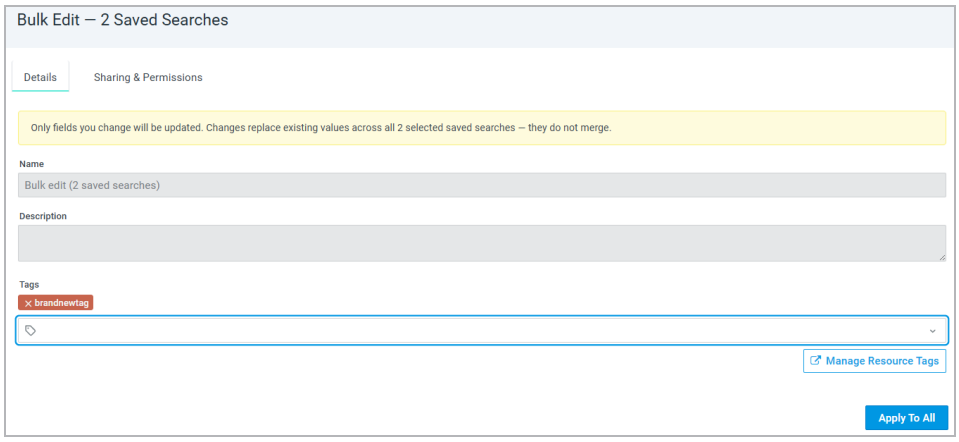
Select multiple saved searches and apply tags, owner, and permissions to all of them at once.



You must own the saved search, or have write permission, to edit its tags and permissions. Organization administrators can bulk edit any saved searches.

To bulk edit tags on saved searches:

1. Navigate to **Search** > **Manage** > **Saved Searches**.
2. Select two or more saved searches. The **Bulk Edit** button appears in the action bar.
3. Click **Bulk Edit** to open the bulk edit slide panel.



4. On the **Details** tab, use the **Tags** field to apply tags to all selected saved searches. The **Name** and **Description** fields are **Read**-only because they are unique per saved search.
5. (Optional) Select the **Sharing & Permissions** tab to update the owner and access control settings for all selected saved searches.
6. Click **Apply to All** to save the changes.

> **Note:** Fields you edit replace the existing values on all selected saved searches; they do not merge. Fields you do not edit are not updated.

Components of a Saved Search

- Select a saved search to load the query in the search bar and issue the search with the same AQL syntax.
- Organization administrators and users with audit privileges can audit saved searches, including the actions of creating, updating, and deleting a saved search.

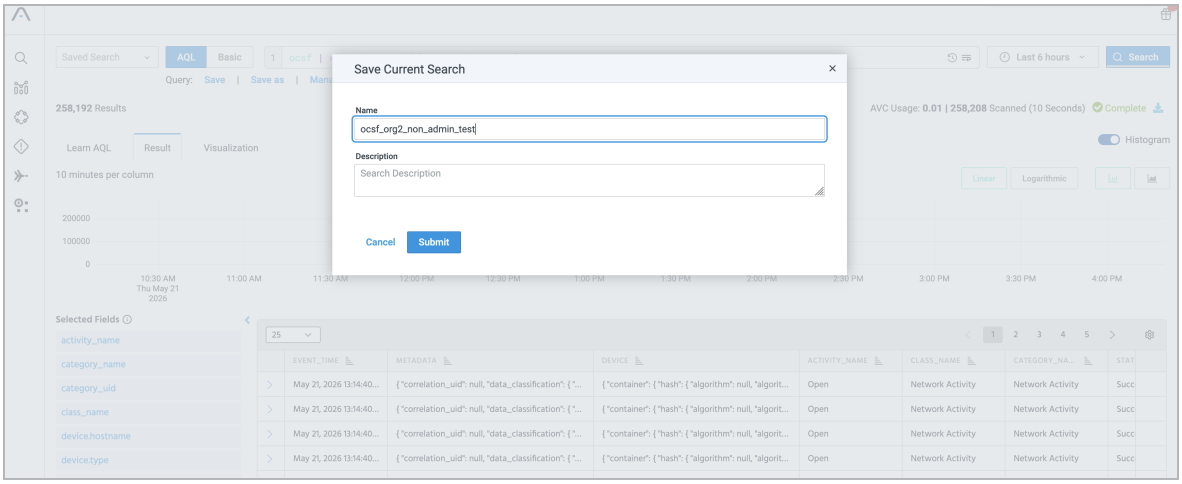
> **Note:** The saved search configuration saves the absolute time range filter that was applied to the search from which it was saved. For example, if you select the relative time period **Today so far** on July 28 at 11:00 AM, issue a query, and save the search, then the absolute time is saved. If you select and run the saved search on July 29, you will get results for July 28 from 0:00

AM to 11:00 AM.

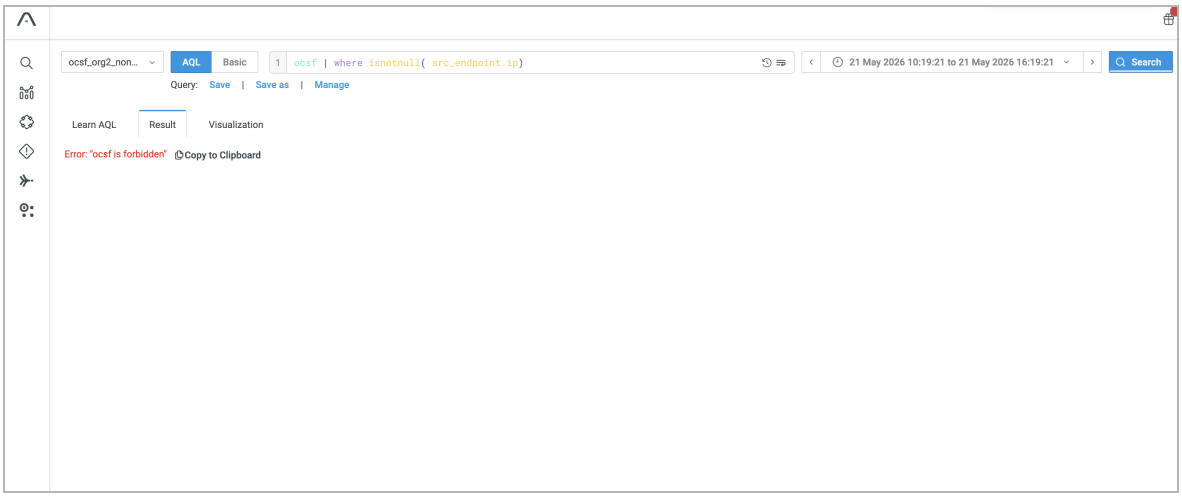
OCSF Saved Searches and Role-Based Access Control

The OCSF filter defined in a user's role controls what data is returned when running an OCSF-based saved search. The following examples illustrate the experience of creating and running an OCSF saved search under different RBAC states.

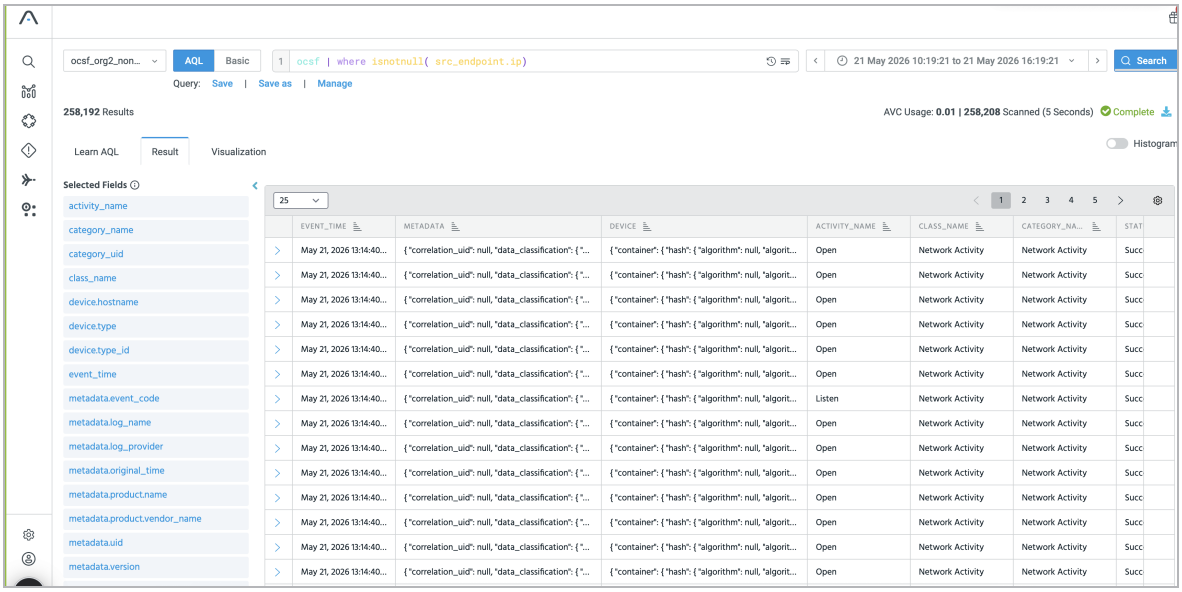
Creating an OCSF saved search: A user with an OCSF role assigned can save an OCSF-based query just like any other search. In the example below, the search `ocsf | where isnotnull(src_endpoint.ip)` is saved as `ocsf_org2_non_admin_test`. The saved search stores the query intact; the OCSF filter from the user's role is applied at run time, not at save time.



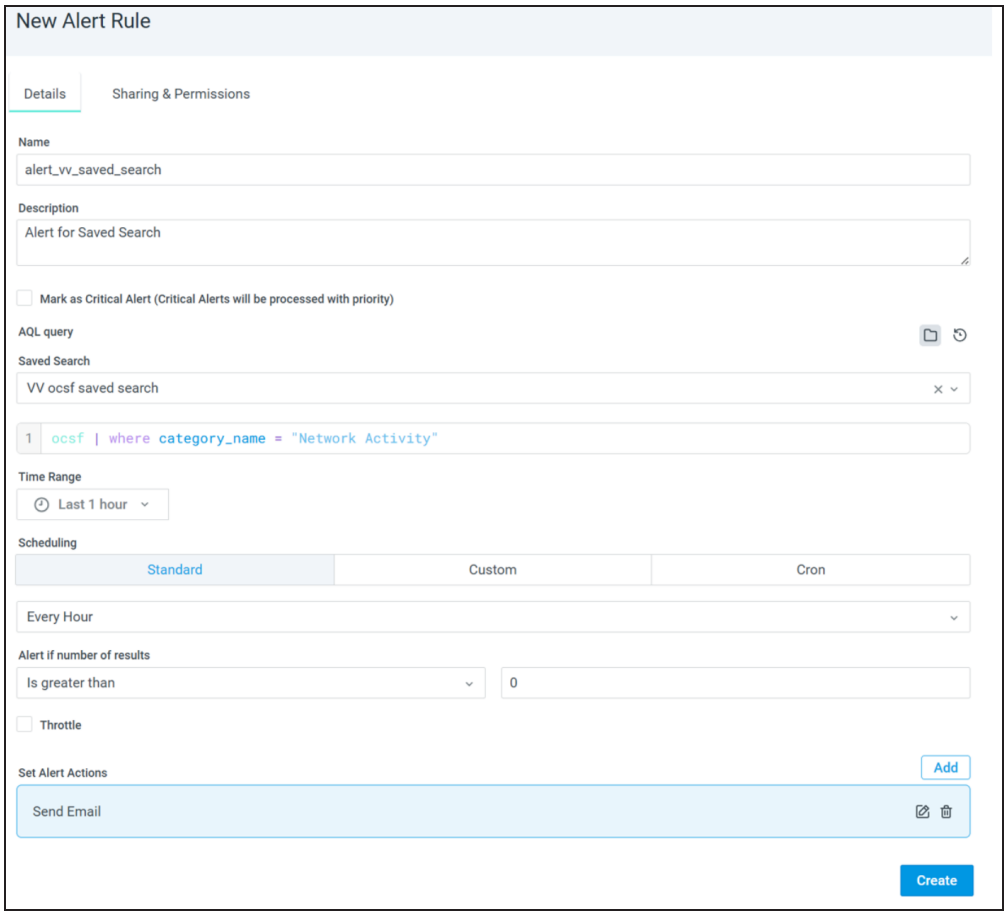
Running an OCSF saved search — OCSF RBAC enforcement not enabled: If OCSF RBAC enforcement is not yet enabled for the organization, non-administrator users receive an `ocsf is forbidden` error when they run an OCSF-based saved search, even if the query itself is valid. The saved search cannot return results until enforcement is enabled and the user has an OCSF role assigned.



Running an OCSF saved search — OCSF RBAC enforcement enabled, OCSF role assigned: After OCSF RBAC enforcement is enabled and the user has a role with an appropriate OCSF filter, running the same saved search returns results filtered to the permitted categories. In the example below, the user's role permits Network Activity, and the saved search returns 258,192 Network Activity events.



Alerts for Saved Searches

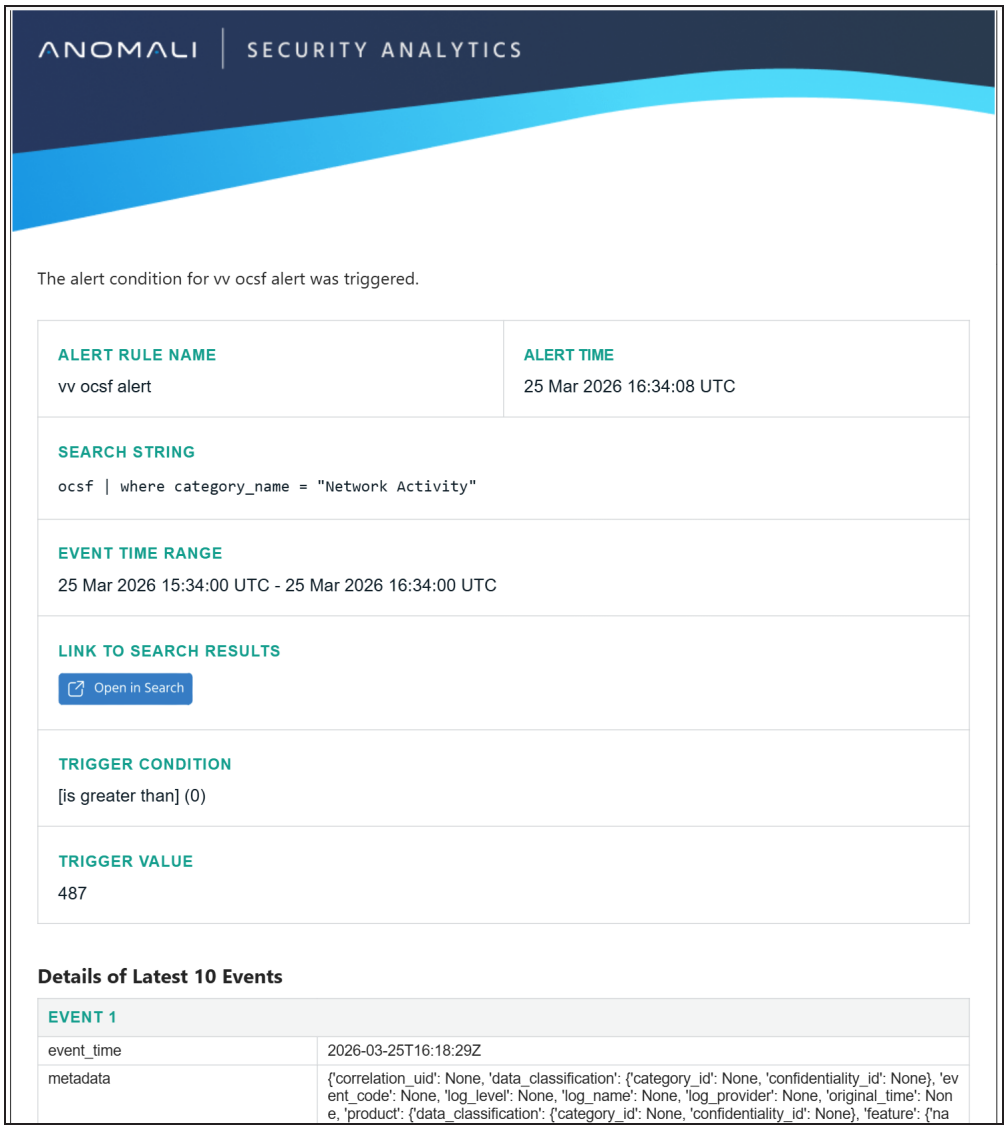


Once you create a saved search, the results table lets you configure an alert with the newly saved search.

To configure an alert from Saved Search:

1. Navigate to **Search** > **Manage** > **Saved Search**.
2. Click the alert bell icon ( ) to open the **New Alert Rule** form.
3. Complete Steps 1–8 of Creating Alerts to finish creating the alert.

In the following example, the OCSF Version 1.2 schema-based saved search named `"vv ocsf alert"` triggers an alert from Anomali Security Analytics that sends an email notification about the saved search, based on how alert was configured.



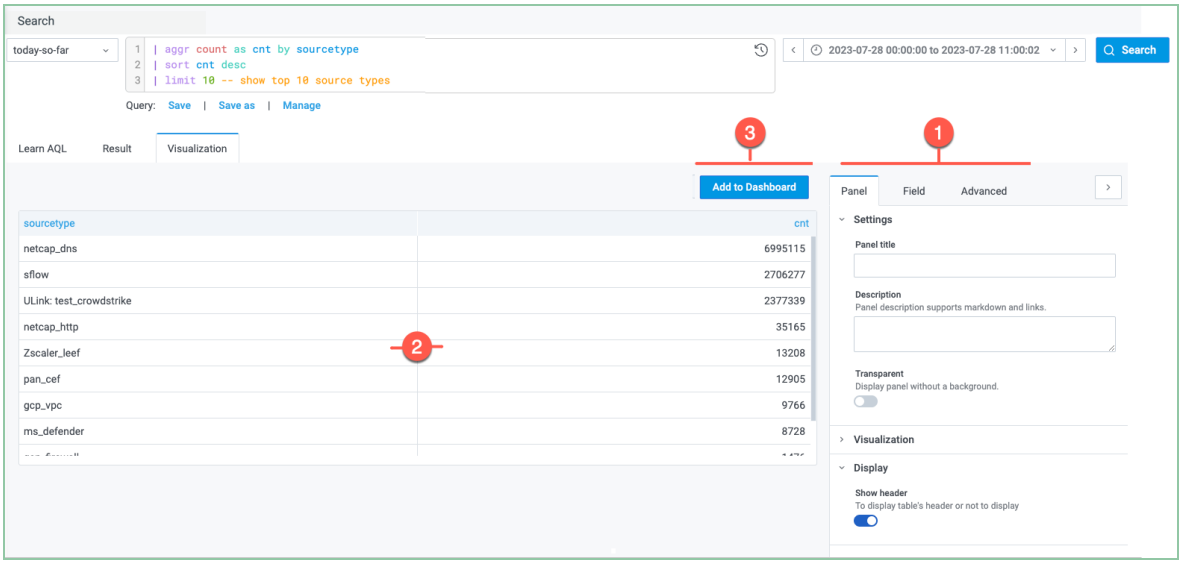
See Anomali Security Analytics Alerts to know more about how alerts automate the process of detecting, notifying, and responding to specific security events that meet user-defined conditions, while minimizing manual oversight.

Working with Visualizations

Visualizations help you gain data insights across different security use cases by providing an interactive visual snapshot of your data. Use the **Visualization** tab to view the associated visualization and configure its properties.

All visualization types support both the eventlog and OCSF Version 1.2 schema fields with dotted notation. See "OCSF Schema Overview" on page 39 for more information on the OCSF fields and naming conventions.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.



1	Configure visualization settings: • **Panel** Use the **Panel** tab to specify a panel title and to select a visualization type and data fields, among other options. See "Panel Settings" on the next page. • **Field** Use the **Field** tab to define options for rendering data fields in the visualization. See "Field Settings" on the next page. • **Advanced**: Use the **Advanced** tab to define options for specific fields that you want to be different from the settings defined on the **Fields** tab. See "Advanced Settings" on page 109.
2	Preview Canvas: The preview area for the visualization. The preview reflects your refined query and applied panel settings.

3	**Add to Dashboard**: After you have customized the visualization as desired, click **Add to Dashboard** and configure the settings to add the visualization as a panel in a new or existing dashboard.

Panel Settings

Use the **Panel** tab to specify a panel title and to select a visualization type and data fields, among other options.

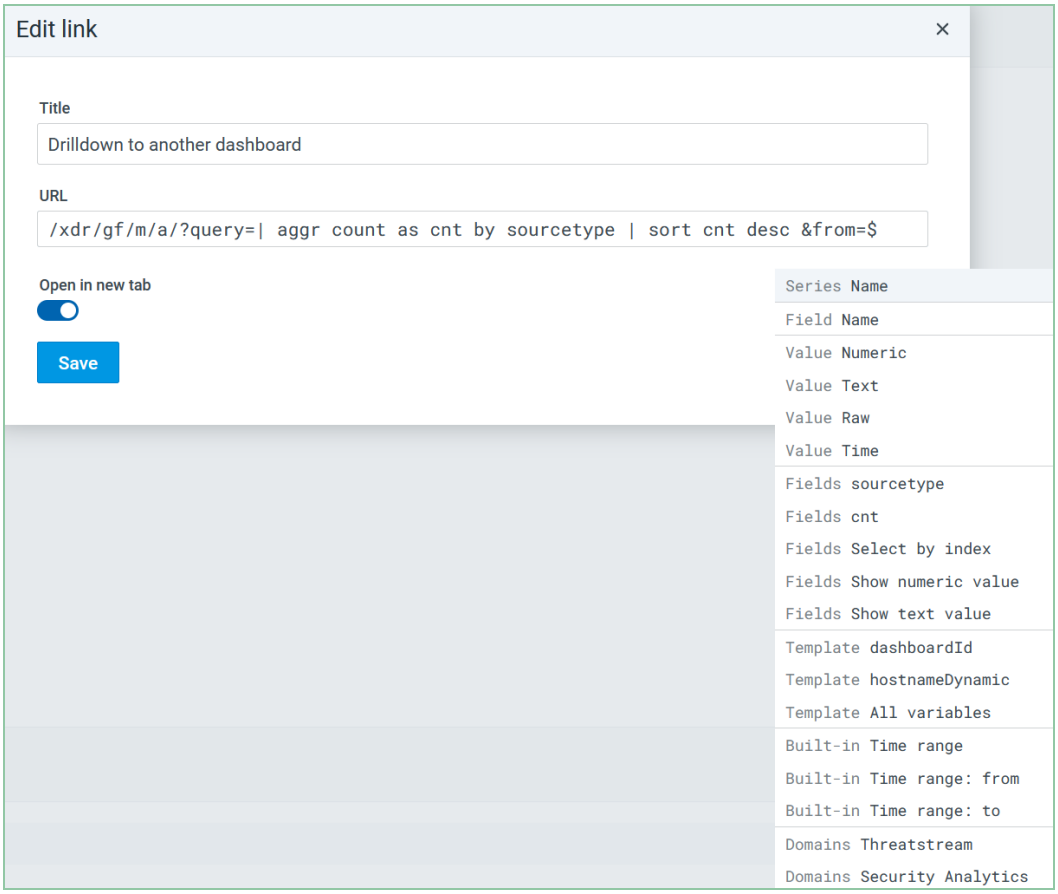
Setting	Description
Panel title	Titles appear at the top of a panel.
Description	Descriptions appear as tooltips. Markdown and links are supported.
Transparent	Display panel without a background.
Visualization	Select a visualization type to display your data. See "About AQL Visualization Types" on page 252 for descriptions of available visualization types. All visualization types support OCSF Version 1.2 fields with dotted notation. See "OCSF Schema Overview" on page 39 for more information.
Display options	Groups of display options vary by visualization type.

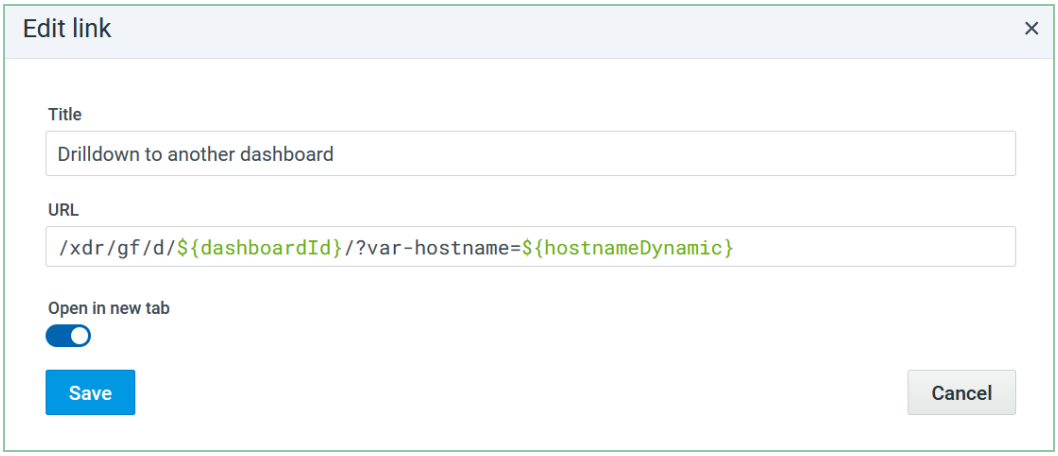
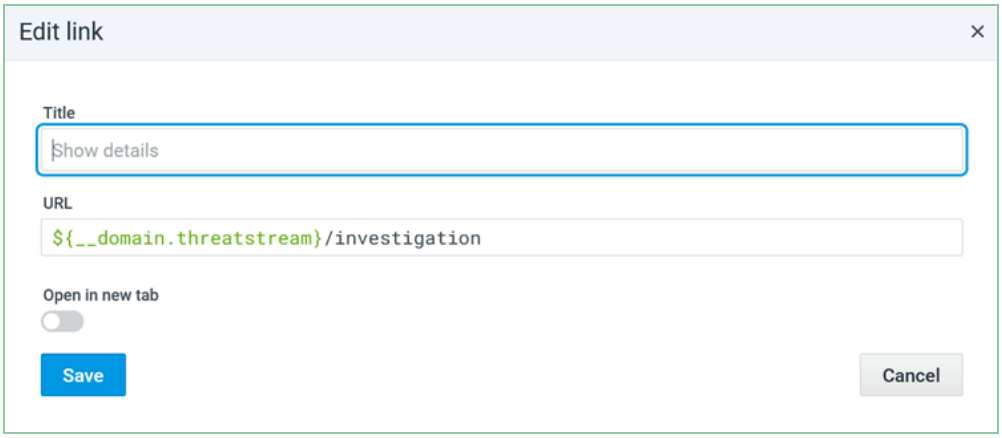
Field Settings

Use the **Field** tab to define options for rendering data fields in the visualization.

Setting	Description
Visualization options	Groups of field settings vary by visualization type.

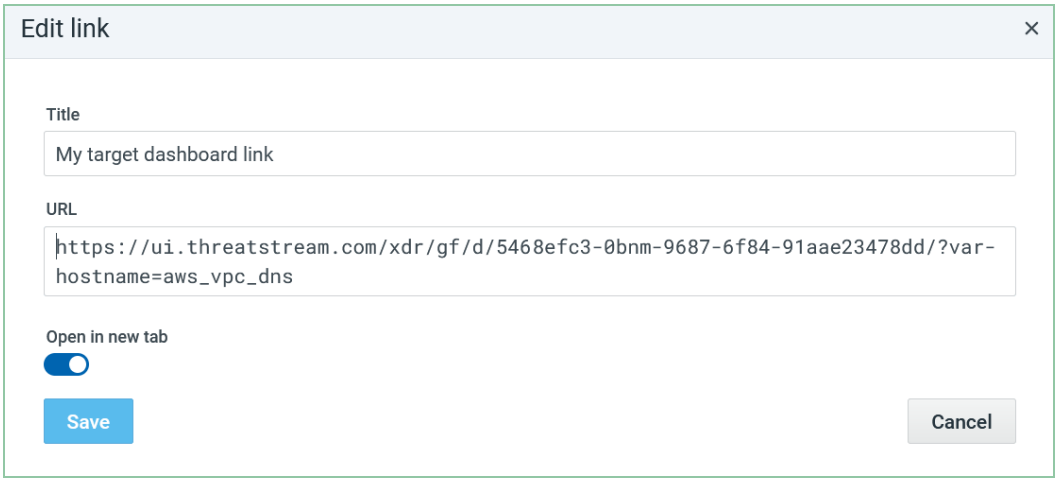
Setting	Description
Standard options	• Unit • Min • Max • Decimals • Display name • No value • Color scheme
Thresholds	If the color scheme is set to **From thresholds (by value)**, specify the colors based on count or percentage thresholds for data fields.
Value mappings	Map a single value or range of values to a specified text string.

Setting	Description
Data links	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query. 

Setting	Description
	• Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.  To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**. 

Setting	Description
	In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream. **Add link to another dashboard** To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting	Description

	`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
	
	 
	
	In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
	
	Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
	
	`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Settings

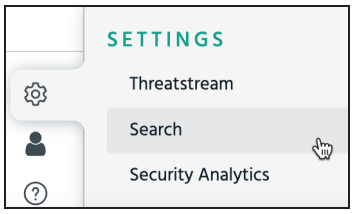
Use the **Advanced** tab to define options for specific fields that are different from the settings defined on the **Fields** tab.

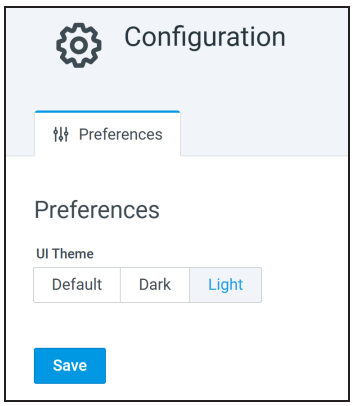
Modifying Search Settings

You can enable the AQL Search auto-complete feature or change display to a dark or light theme for the Search and Dashboards app pages. The preference is saved per user.

To modify preferences:

1. In the left navigation panel, navigate to **Settings** > **Search**.
   This page displays the Preferences.





2. Change to your preference.

   a. UI Theme: Select a theme for the AQL Search and Dashboards.

3. Click **Save**.

Chapter 3: Role-Based Access Control for Schemas

Security operations teams are often organized by domain, with some teams focusing on network activity, others on application behavior, or system events. Role-Based Access Control (RBAC) for schemas helps administrators ensure that data access for each user aligns with their function by assigning roles with targeted filters. This access ensures users see only what their work requires and administrators have confidence that access is both accurate and appropriately scoped.

RBAC in Anomali Search lets organization administrators control which event data each user can access. RBAC now supports the "OCSF Schema Overview" on page 39 in addition to the standard Eventlog schema, allowing administrators to restrict user access to specific OCSF categories.

This page covers the schema-specific aspects of RBAC. For foundational information on roles, user types, permissions, and general role management, see User and Role Management on the Anomali Platform.

About Role-Based Access Control

Roles define the permissions and data access filters that apply to users assigned those roles. On the Anomali platform, organization administrators manage roles from **Settings** > **User & Role Management**.

Key principles of RBAC relevant to OCSF:

- **Organization Administrators**: Bypass all RBAC filters and have unrestricted access to all schemas and data.
- **Non-administrator users**: Use permissions according to the filters defined in the roles assigned to them. If no filter is set in a role, that role does not restrict the data the user can access.
- **Multiple roles**: When a user is assigned more than one role, their final permissions are the union of all the assigned roles, that is, their role permissions are combined using OR logic. The user can, therefore, access any data permitted by any of their assigned roles.

- **Schema-specific filters**: Eventlog filters and OCSF filters operate independently. An eventlog filter on a role restricts access to the standard eventlog data. On the other hand, an OCSF filter restricts access to OCSF data.
- **One filter per role**: A role can only have one filter, either an eventlog or OCSF filter, but not both simultaneously. A user can be assigned multiple roles, with one role containing an eventlog filter and another role containing an OCSF filter. However, a user cannot be assigned a role that has multiple filters.

Schema Filter in Roles

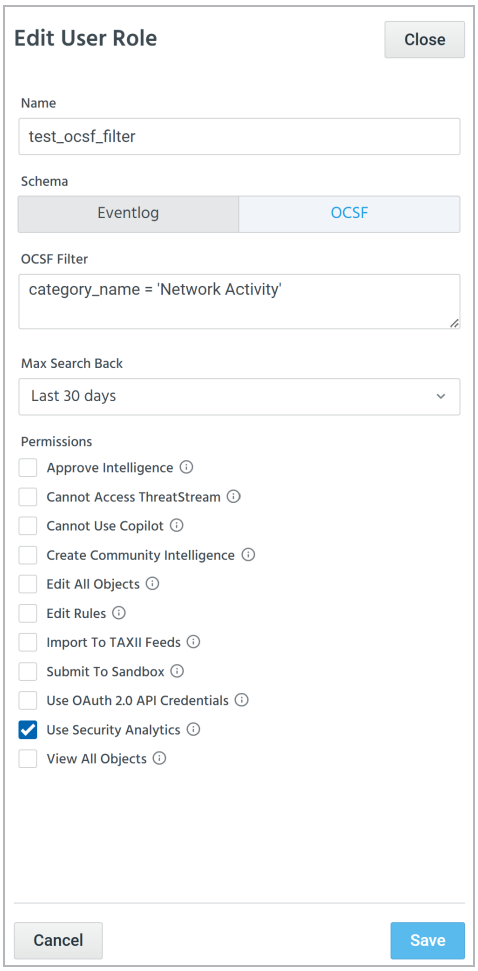
The role creation and editing form includes a **Schema** selector with two options: **eventlog** and **OCSF**.

- Select **eventlog** to define an eventlog-based filter in the **Filter** field. Use this field to restrict access to eventlog data using fields such as `sourcetype`, `event_tags`, `ext_source`, and `ext_sourcetype`. For example: `sourcetype in ('aws_vpc_flow')`.
- Select **OCSF** to define an `ocsf`-based filter in the **Filter** field. Use this field to restrict access to OCSF data by specifying a `category_name` constraint.

  Example: `category_name = 'Network Activity'`

A role supports only one filter type. Switching the schema selector clears any filter previously entered for the other schema type.

For the list of supported OCSF categories, see "Supported OCSF Categories" on page 41.



Note: A Security Analytics subscription is required to use the OCSF Filter field.

Configuring a Schema Filter for a Role

To configure an OCSF filter for a role:

1. In the bottom-left corner of the side navigation panel, click **Settings** > **User & Role Management**.
2. Click the **Roles** tab.
3. Select an existing role to edit, or click **New** to create a new role.

4. (Optional) In the **Schema** selector, click **OCSF** to define the **OCSF Filter**. If you select **Eventlog**, the field name becomes **Eventlog Filter**.

   A role supports only one filter type. Switching the schema selector clears any filter previously entered for the other schema type.

5. In the **OCSF Filter** field, enter a `category_name` constraint to restrict user access to the desired OCSF category. For example:

   ```
   category_name = 'Network Activity'
   ```

6. (Optional) Select the **Max Search Back** range and any **Permissions** to include in this role.

7. Click **Save**.

The role is saved with the OCSF filter applied.

> **Note:** If a role includes an invalid OCSF constraint, a validation error is returned when saving. Ensure the constraint uses the correct `category_name` syntax and a supported OCSF category value.

How the Schema Filter Works

The following table illustrates how role assignment and schema (OCSF or eventlog) filter affect search results, covering both unrestricted and filtered scenarios.

In this table, "Unrestricted" means the query runs without any data filter applied, and returns all results matching the query.

Use Case	Role Configuration	OCSF Query Result	Eventlog Query Result
1	No role assigned	Unrestricted	Unrestricted
2	Role assigned, no filter defined	Unrestricted	Unrestricted

Use Case	Role Configuration	OCSF Query Result	Eventlog Query Result
3	Role assigned with an eventlog filter, no OCSF filter	`ocsf is forbidden`	Unrestricted
4	Role assigned with OCSF filter: single category	Filtered to that category	`eventlog is forbidden`
5	Role assigned with OCSF filter: multiple categories	Filtered to permitted categories	`eventlog is forbidden`
6	Role assigned with OCSF filter only, user runs eventlog query	Filtered as per role	`eventlog is forbidden`

Use Case 1: Non-Administrator with No Role Assigned

When a non-administrator user has no role assigned, OCSF RBAC enforcement does not apply. Both eventlog and OCSF queries run without restriction, regardless of whether OCSF RBAC is active for your organization.

Use Case 2: Non-Administrator with a Role Assigned but No Filter Defined

When a non-administrator user has a role assigned but the role does not include an eventlog or OCSF filter, both eventlog and OCSF queries run without restriction. The system enforces only the filter that is explicitly defined in an assigned role. If no filter is set, no restrictions apply.

Use Case 3: Non-Administrator with a Role Assigned but No OCSF Filter

When OCSF RBAC is active and a non-administrator user has a role assigned that does not include an OCSF filter, all OCSF queries return an `ocsf is forbidden` error. The user cannot access any OCSF data until an organization administrator assigns them a role that includes an OCSF filter.

Use Case 4: Non-Administrator with a Single-Category OCSF Role

When a non-administrator user is assigned a role with an OCSF filter of `category_name = 'Network Activity'`, Anomali Search automatically appends the filter to

every OCSF query in the backend. The user does not need to include the filter in the search bar, and the filter does not appear in the query as typed.

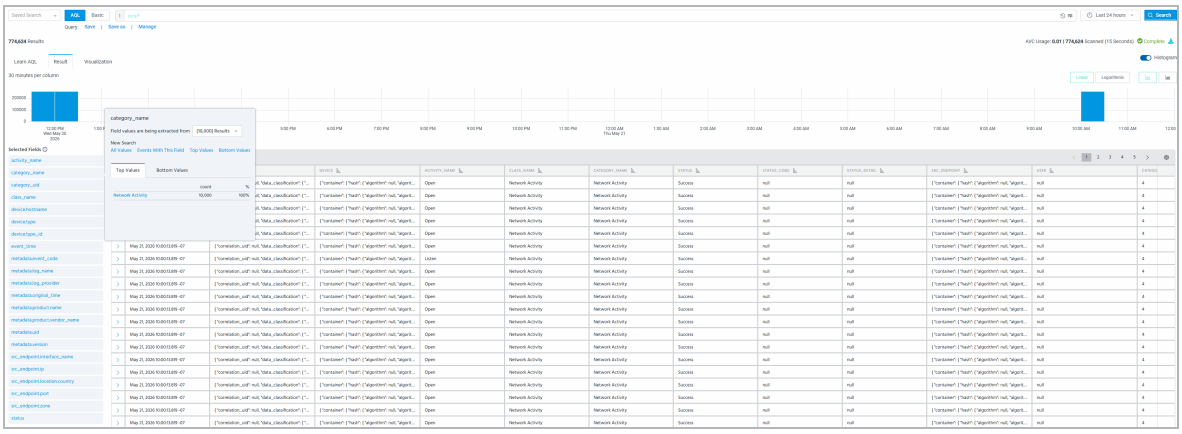
For example, the following user query

```
ocsf
```

is automatically updated by Anomali Search to execute the equivalent of:

```
ocsf | where category_name = 'Network Activity'
```

The constraint is applied silently and automatically based on the role, so the results contain only `Network Activity` events regardless of how broadly the user queries.



Use Case 5: Non-Administrator with a Multi-Category OCSF-filter Role

When a non-administrator user is assigned a role with an OCSF filter that permits multiple categories — for example, both `category_name = 'Network Activity'` and `category_name = 'System Activity'` — the filters are combined using OR logic to include both these OCSF categories. Therefore, the user can access events from both these permitted categories.

In the example below, running

```
ocsf | aggr count as cnt by category_name
```

returns results for both the categories permitted by the filter in the assigned role.

Use Case 6: User Querying Eventlog Data with OCSF-Only Filter

OCSF and eventlog access are controlled independently. A user whose role grants access to OCSF data but does not include an eventlog filter cannot query eventlog data. If such a user runs an eventlog query, Anomali Search returns a forbidden error, because the underlying query targets the eventlog schema that the role prohibits.

This filter is bidirectional:

- an eventlog filter on a role does not grant OCSF access
- an OCSF filter on a role does not grant eventlog access.

Therefore, users who need to query both schemas must have appropriate roles configured for each schema: one role containing an eventlog filter, and another role containing an OCSF filter.

Schema Filter Scope Across Search Features

Schema filters defined in user roles apply across all Search platform features, not only to direct AQL queries. This includes macros, Copilot, saved searches, visualizations, reports, views, full-text indexing, turbosearch, and correlation lookup tables. In every case, the same access rules apply to both schemas:

- Eventlog filters restrict what eventlog data is returned when any feature runs an eventlog query. A user without an eventlog-filter role will receive an `eventlog is forbidden` error when a feature attempts to run an eventlog query.

- OCSF filters restrict what OCSF data is returned when any feature runs an OCSF query. A user without an OCSF-filter role will receive an `ocsf is forbidden` error when a feature attempts to run an OCSF query.

See "Saved Searches" on page 88 for an example of how Schema RBAC enforcement works in the saved search workflow. See "OCSF Field Panel" on page 80 for a search feature that uses OCSF category context.

Schema-RBAC Enforcement

Schema RBAC enforcement is an optional capability that controls whether data access is restricted by the schema filters defined in user roles. This applies equally to both the eventlog and OCSF schemas.

Contact your Anomali administrator or Anomali Support to activate Schema-RBAC for your organization.

Current Behavior

The following behavior applies by default, before Schema-RBAC is active for your organization:

User Type	Query Behavior
Organization Administrator	All queries run without restriction.
Non-administrator	OCSF queries such as: • `ocsf` • `ocsf | turbosearch 1.2.3.4` • `dgamatch_ocsf` run without any OCSF filter applied. Similarly, standard `eventlog` filters continue to apply to eventlog queries.
User without Security Analytics access	OCSF queries return a Forbidden error. Access to allowed tables such as `observables` and `threat_models` is unaffected.

After Schema-RBAC is Active

See "How the Schema Filter Works" on page 113 for a detailed breakdown of how queries behave under each role configuration when Schema-RBAC is active.

Chapter 4: Full-Text Log Indexing

> **Note:** This is a limited-availability feature. Contact Anomali Customer Support for more information.

Anomali Search offers full-text indexing on eventlog data, a type of search capability that allows you to directly perform a keyword search on the raw message and ensures faster search results, while reducing the AVC units you consume.

Full-text indexing in Anomali is near real-time, with most events indexed within 10 minutes after ingestion.

Key Technical Guidelines

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Full-text indexing entails the following technical highlights:

- Full-text indexing is applicable to both the `eventlog` and `ocsf` schemas. See "OCSF Schema Overview" on page 39 for more information on OCSF Version 1.2 parsing and naming conventions.
- You must place the desired keyword at the beginning of the query to leverage full-text indexing. To use the OCSF schema, add the case-insensitive schema name "`ocsf`" as a prefix before the desired keyword.
- You must enclose the desired keyword in double quotes ("") to leverage full-text indexing.
- Full-text indexing does not support wildcards, partial matches, or substrings. Only exact matches are supported. This applies to sourcetype naming as well.

- All full-text searches ignore character case. For example, `"Error"` and `"error"` are treated identically.
- For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.
- The full-text indexer performs tokenization using the following definitions:
  - Valid characters: letters, digits, dot `(.)`, dashes `(-)`, underscore `(_)`.
  - Delimiters: spaces, punctuations, symbols, special characters.
- Full-text indexing is supported by AQL search queries across the Anomali platform.

How to Use

Example 1: Enter a desired keyword enclosed in double quotes (`""`), just like you would in a Google-like search.

```
"john.doe@acmecorp.com"
```

Example 2: As an example, combine AQL operators on the raw message, such as the where operator, with the desired keyword at the start of the query.

```
"Alice.Bob" | where sourcetype="windows_logs"
```

```
"Alice.Bob" | aggr count as total_count by sourcetype | sort cnt desc | limit 10
```

Example 3: The desired keyword must be at the start of a query. Here, the email address is after the `where` operator, so full-text indexing is not applied.

```
| where message contains "John.Doe@acmecorp.com"
```

Example 4: Full-text indexing applies to keywords at the start. Here, `"abcd"` will utilize full-text indexing, but not `"efgh"`.

```
"abcd" | where message contains "efgh" | limit 10
```

Example 5: Full-text indexing applies to the `ocsf` schema. Here, the indexer looks for a UUID in an ingested log.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

```
ocsf "d3590ed6-52b3-4102-aeff-aad2292ab01c" | aggr count as cnt by category_name
```

Example 6: Wildcard notation or substring searches are not allowed:

- Search keyword `"acme"` will not return results when your log data contains `"john.doe@acmecorp.com"`.

Chapter 5: Search Data Retention Policy

By default, Anomali Search retains event data in searchable "hot" buckets for 90 days. After this period, data automatically moves to archive buckets, where it is preserved according to the retention terms defined in your license agreement.

Contact Anomali Support for more assistance with retrieving the archived data.

Searching Archived Data

Queries against archive buckets may result in significantly higher Anomali Virtual Compute (AVC) usage. To safeguard against unintentional resource drain and AVC usage, searches are restricted to hot buckets by default.

If your use case requires access to archived data, Anomali recommends submitting a support ticket. Once approved, your user account will be configured to search beyond the default window.

See Anomali Virtual Compute to understand its usage and the different user actions that lead to AVC usage.

Chapter 6: Anomali Virtual Compute

Anomali Virtual Compute (AVC) is a computing framework that measures your usage of virtualized infrastructure whenever you perform a compute operation.

Operational Scope

Any search operations you perform on the Anomali platform counts towards your AVC limit. These operations include:

- Search queries through the Search UI
- Dashboards queries
- Scheduled jobs
- Alerts
- Search APIs

Target audience

Any user with access to Search and Dashboards can see their AVC usage on the Anomali platform.

Monitoring usage

See Monitor your AVC usage to get started on using the AVC framework to understand and monitor your search usage and usage behavior. Furthermore, see Best Practices to learn how you can leverage AVC for optimal resource usage and optimization.

Best Practices for Managing Usage

Anomali recommends the following guidelines to optimize AVC usage and reduce overhead.

Shorter Search Time Ranges

Select shorter time ranges to minimize data processing, such as 24 hours instead of longer ranges such as 6 months or 1 year.

Here are example scenarios for low- and high-AVC searches:

Low AVC search example: Targeted shorter duration searches that employ Turbosearch on indexed fields and filtering or aggregation on parsed fields.

High AVC search example: Searches with longer duration involving heavy data ingestion or searches on raw messages with the `json_extract` or regular expression functions in the `calc` operator. See the calc operator for more information.

Utilize Parsed Fields

Prioritize parsed fields over searches for raw messages to ensure higher operational efficiency.

High AVC approach: `| where raw_message contains_ci ("John")`

Recommended approach: `user = "John"`

Incorporate Turbosearch

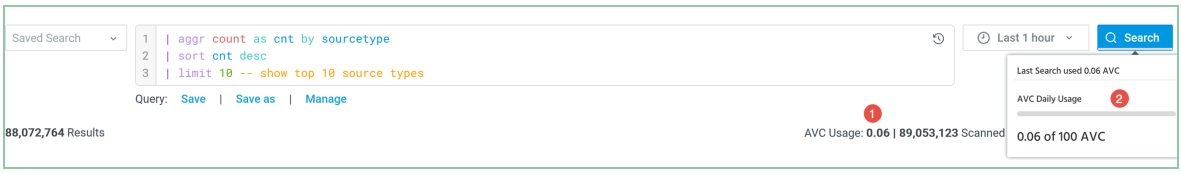
Use the Turbosearch operator as the initial Search operator whenever feasible. Here are some examples:

```
|turbosearch "https://www.example-url.com/feed"
```

```
|turbosearch "sender_example@doc*" "examplefilehash" "1.2.3.4"|
aggr count (sourcetype) as srctype_count
```

Monitor your AVC usage

In Anomali Search, you can monitor your AVC usage from the Search homepage.



1	AVC usage: View how many AVC units your current query consumed.

2	Search (Search) button: Hover the mouse on the **Search** ( ) button to view how many AVC units usage. In this dialog, you can: • View the total AVC units consumed from your daily quota and the remaining balance for today. As an example, in the screenshot, the user's search query has consumed 0.06 units of their daily 100 AVC units. • Monitor the status of your AVC units, represented by different colors: ▪ White: AVC consumption is within limits. ▪ Yellow: AVC limit is approaching the allocated quota for the day. ▪ Red: AVC limit for the day has been reached for the user.

Chapter 7: Using Lookup Tables

A *lookup table* is a CSV file that has enrichment data and can be any set of fields that you want to correlate with an event log field, enabling the lookup table data to enrich Search results.

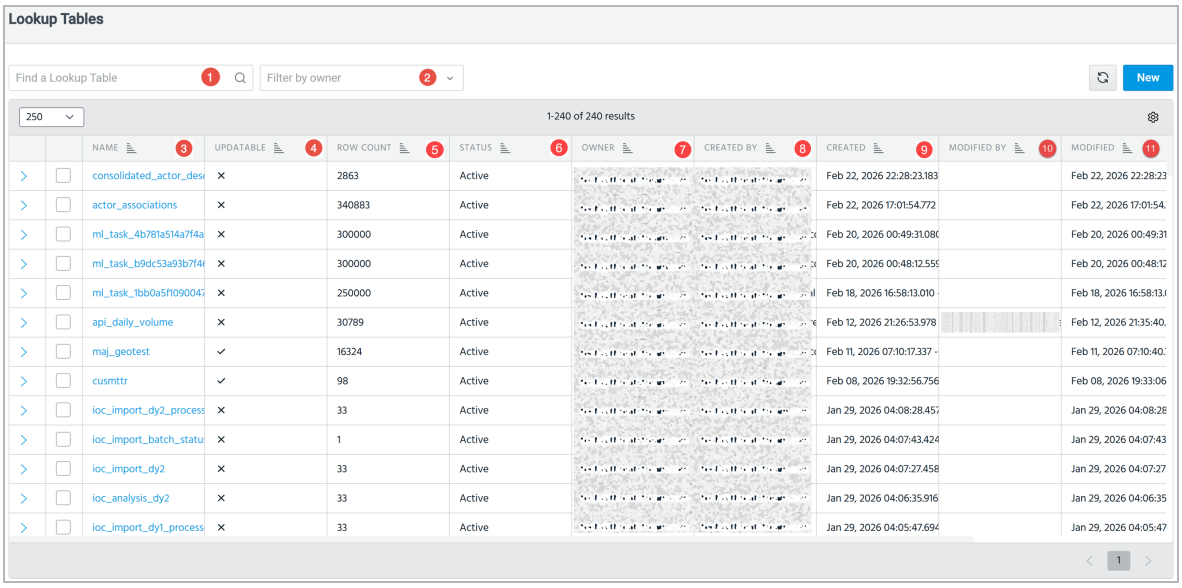
For example, the eventlog lookup table empowers AQL Views and Security Analytics to map Anomali's event log schema to different event fields from varied event log sources.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Anomali offers and supports two types of lookup tables:

- **System lookup tables:** These are lookup tables that Anomali offers out of the box. See "Anomali System Lookup Tables" on page 131 for more information.
- **Custom lookup tables:** Anomali allows you to create your own lookup tables for downstream applications.
  To learn how to create a custom lookup table, refer to "Creating Custom Lookup Tables " on page 136.
  To learn how to view, append, update, export, or delete custom lookup tables, refer to "Managing Custom Lookup Tables" on page 140.

To access all lookup tables available to your organization, navigate to **Search** > **Lookup Tables**.

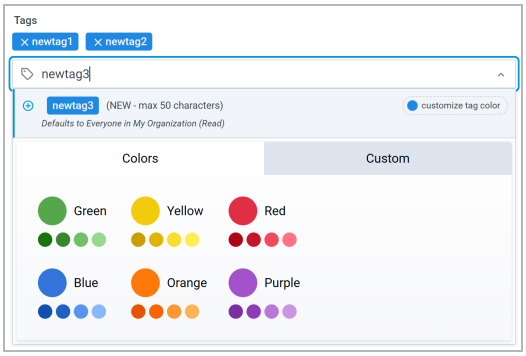


1	**Search:** Enter the lookup table name that you want to find. You can also search by name, description, modified by, created by, or owner, with the search being incremental, that is, results update as you type.
2	**Filter by owner**: You can filter lookup tables by owner. Possible values include: • **Mine**: returns tables for which you are the owner. • **System**: returns out-of-the-box tables, that is, tables that are owned by nobody and that cannot be changed. • **Not System**: returns tables that are not owned by system, such as tables created by a user. • **<user>**: returns tables where the selected user is the owner. **Note:** A user can always only view and edit tables they own. See "User Sharing and Permissions" on page 130 for more information.

	Filter by tag: Filter the lookup table list by one or more resource tags. • The drop-down shows only tags you own or that are shared to your organization. Organization administrators can also see private tags owned by other users. • Selecting multiple tags returns only lookup tables that have all selected tags applied. • Click **Clear tags** to remove the filter. • The count next to each tag shows how many lookup tables have that tag applied, not the total across all the supported resource types. See "Managing Custom Lookup Tables" on page 140 for details.
3	**Name:** Name of the lookup table.

Tags: Apply one or more resource tags to the lookup table.

- The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.
- To apply an existing tag, type in the field and select it from the auto-complete suggestion results.
- To create a new tag, type a name that does not match any existing tag and confirm the selection. A color picker appears so you can choose a tag color before the tag is created.



- After selecting the color, click the add icon (⊕) to add the tag to the lookup table as well as the shared tag catalog. A new tag is added to the shared tag list immediately, even if you abandon creating or editing the lookup table.

- New tags default to **Everyone in My Organization (Read)**. You can change this later from the "Resource Tags" on page 146 settings page
- Applied tags render as colored chips. Click the cross icon (×) on the tag chip to remove the tag from the lookup table. This does not delete the tag from the catalog. Any user with edit

	access to the saved search and visibility to a tag can remove the tag. • To filter the list by tag, see "Managing Custom Lookup Tables" on page 140. **Note:** Users with read-only permission on a lookup table can view applied tags but cannot add or remove them. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.
4	**Updatable:** Whether the lookup table is updatable.
5	**Row Count:** Number of rows in the table.
6	**Status:** View the activation status of the lookup table: • Active: An active status indicates that the table has been created and is ready to be used in Search. You cannot activate a lookup table. • Inactive: An inactive status indicates that activation is not yet completed.
7	**Owner**: Indicates the owner of the lookup table. By default, the creating user is the owner and is private (not shared). The owner can: • Modify all lookup table details. • Configure sharing or permissions granted to users. • Edit can delete the table. **Note:** A user can always only view and edit tables they own. See "User Sharing and Permissions" on the next page for more information.

8	**Created By:** Email address of the user that created the lookup table. A user can always only view and edit tables they own.
9	**Created:** Denotes the timestamp of when the table was created.
10	**Modified By:** Refresh the list of lookup tables.
11	**Modified:** Denotes the timestamp of when the table was modified.
12	**New:** Create a new lookup table. Refer to "Creating Custom Lookup Tables " on page 136 for details.
	(Optional) Enter a description for the lookup table.

User Sharing and Permissions

All users can always view and edit lookup tables they own. They can see only lookup tables they are entitled to view, based on the permissions granted under the Sharing & Permissions tab.

If the Sharing & Permissions settings are configured as:

- **Private**: Only the owner of the lookup table can view and edit the table. By default, if no sharing is specified, the lookup table is private.
- **Shared**: Users with specified roles can view and edit the lookup table, according to their assigned permissions

Users with roles granted read permissions:

- Can view, clone, and export lookup tables they own. Can only view, and cannot edit, the tables they do not own.

Users with roles granted write permissions:

- Can modify the lookup table, but not sharing or permission rights.
- Can use the "yieldtable" on page 719 operator, as it supports ownership and sharing parameters.

Users who own the lookup table:

- Can perform all actions, including editing ownership and permissions of the dashboards they own.
- If no sharing is specified, by default, the lookup table is private (not shared).

Organization Administrators

Can perform all actions, including editing ownership and permissions of any dashboard.

Referencing Lookup Tables in AQL Search:

- Only visible lookup tables can be referenced.
- Auto-complete feature does not expose lookup tables to which you lack access.
- Structured search also does not expose lookup tables to which you lack access.
- Lookup tables can be joined with both eventlog and OCSF queries. See "OCSF Schema Overview" on page 39 for more information about OCSF queries.

Example: Joining OCSF Data with a Lookup Table

You can enrich OCSF queries by joining them with lookup tables. The following example joins OCSF network activity data with a custom lookup table containing known malicious IPs:

```
| from malicious_ips
| join type=inner (
    | from eventlog
    | where event_type = "network"
) on src_ip = ip
| fields event_time, src_ip, dest_ip, threat_category, confidence
```

Anomali System Lookup Tables

The table below summarizes the lookup tables that Anomali provides out of the box.

Source	Lookup Table	Description
Your event logs	"eventlog " on page 746	Represents the schema that Security Analytics uses to map to different event log fields from varied event log sources. As an example, here are some categories of essential fields captured by the Eventlog Schema, among several others: **Network:** `event_time, action, src_ip, dest_ip, src_port, dest_port, protocol, bytes_in, bytes_out` **Web Proxy:** `event_time, action, src_ip, dest, url, user, user_agent, http_referrer, status, http_method` **DNS:** `event_time, action, src_ip, query, answer, record_type` **Email Gateway:** `event_time, message_id, subject, sender, receiver, src_ip, dest_ip` **EDR:** `event_time, event_id, event_name, evt_type, host, user, platform, process_id, file_path, file_name, image, command_line, file_hash, file_hash2, file_hash3, src`

Source	Lookup Table	Description
Security Analytics	"iocmatch" on page 1281	Generated by the Anomali Security Analytics IOC correlation engine. This engine correlates event telemetry — firewalls, proxy servers, email gateways, or endpoints, and more — with ThreatStream IOCs. The following fields are used for correlating events: **IP addresses:** `src_ip`, `dest_ip`, `src`, `dest` **Domain:** `src`, `dest`, `query`, `url` **URL:** `url`, `http_referrer` **Email:** `sender`, `receiver` **File Hash:** `file_hash, file_hash2, file_hash3`
Security Analytics	"dgamatch" on page 1255	Generated by the Anomali Security Analytics domain generated algorithm detection feature. This feature correlates domain names found in your event telemetry data with Anomali DGA intelligence.
Security Analytics	"ignite" on page 1267	Generated by the Anomali Ignite detection feature that correlates file hashes present in event logs with endpoint file hashes that identify high-fidelity intelligence.
Security Analytics	"tmmatch" on page 1305	Generated by threat model correlation that summarizes threat model matches present in your event logs.

Source	Lookup Table	Description
ThreatStream	"observables" on page 1298	Contains observable details generated from several sources such as intelligence feeds, telemetry data (SIEM, EDR data, and so on), threat bulletins, STIX/TAXII servers, and more. This lookup table features a centralized repo of threat observables from ThreatStream that supports correlation, enrichment, analysis, and integrations.
ThreatStream	"threat_models" on page 1306	Contains details about Threat Models in ThreatStream. This lookup table enables correlation and threat hunting, curates threat profiles, aiding in response prioritization, reporting, and defense planning.
Security Analytics	audit	Generates different user actions such as creating alerts, launching a forensic search, or creating a dashboard, among other actions. This lookup table helps you gain crucial insight into the history of different user actions, ensuring transparency, traceability, compliance, reporting, and accountability for every action initiated by API clients, users, or other automated processes.

Source	Lookup Table	Description
ThreatStream	"consolidated_actor_descriptions_ng" on page 1241	Adds AI-generated descriptions to the AI Actor Profile dashboard for each unique threat actor and its aliases in ThreatStream.
ThreatStream	"consolidated_actor_observables_ng" on page 1245	Links threat actor associations from the `consolidated_actor_profiles_ng` lookup table with corresponding observable information based on Actor IDs. This table contributes data to the AI Actor Profile dashboard.
ThreatStream	"consolidated_actor_profiles_ng" on page 1250	Populates the AI Actor Profile dashboard with consolidated information about threat actors, including aliases, associations, motivations, and other contextual details available in ThreatStream.
ThreatStream	"actor_associations" on page 722	Generates information about threat actor associations available in your ThreatStream instance.
Security Analytics	"asset" on page 725	Maintains a continuously updated reference that stores metadata of imported assets such as hostnames, IPs, ownership, criticality, or location, and is automatically refreshed whenever new asset data is ingested. This asset data enriches search results with asset context for more accurate monitoring and reporting.

Source	Lookup Table	Description
ThreatStream	"investigation" on page 733	Stores metadata for security investigations created in ThreatStream, including assignment, priority, status, Traffic Light Protocol, and timestamps. This table supports querying and reporting on investigation workflows across your organization.
Security Analytics	"investigation_elements" on page 735	Provides a centralized reference for all investigation-related elements used within Security Analytics. Enriches investigations with full entity data from the underlying threat model or observable. This table standardizes and organizes investigation data to support consistent querying, reporting, and analysis.

Creating Custom Lookup Tables

You can add a lookup table to your AQL Search environment by uploading a file in CSV, .gz, or .tar formats. Additionally, you can add values to existing lookup tables.

CSV Import Guidelines

To successfully upload a lookup table, follow these file upload guidelines:

- The upload file must have a heading row with the lookup table field names.
- Avoid using the same field names as the eventlog schema.
- In searches, field name is case sensitive, so use capitalization conventions that are easy for search users to follow. For example, all lowercase.
- Enclose values that have spaces in straight quotation marks (").

- File formats must be CSV, .gz, or .tar.

> **Note:** If a column in your CSV file contains IP addresses in CIDR notation (for example, `10.0.0.0/24`), the platform automatically adds two additional columns for each such column in the CSV to the lookup table : `<column_name>_start` and `<column_name>_end`.
>
> For example, if `IPV4_address` is the column name in your CSV file, the platform adds the following columns: `IPV4_address_start` and `IPV4_address_end`.
>
> These columns are used internally to optimize lookup performance and are generated automatically. You can safely ignore them.

- The upload file can be a maximum of 100 MB.

> **Note:** If your CSV file is larger than 100 MB, use the following guidelines to upload your data:
>
> - Compress the file to reduce its size.
> - Split the content into multiple smaller files. After that, upload the first file to create a lookup table and append the remaining files to this table. See "Appending to an Existing Lookup Table" on page 140 for more information.

The following is an example of a lookup table:

	A	B	C	D	E
1	geo_ip	hostname	country	city	zipcode
2	192.0.2.1	tn-1-host	CA	Vancouver	"V6H 3K1"
3	198.51.100.1	tn-2-host	US	San Francisco	94117
4	203.0.113.0	tn-3-host	MX	Guanajuato	36050
5					

To upload a lookup table:

1. Create a CSV file that contains the required values, as described above.
2. Navigate to **Manage** > **Lookup Tables**.
3. Click **New**.

4. Enter a table name in the **Name** field.

> **Note:**
>
> - This is the lookup table name that users can reference in AQL Search.
> - The name may contain only alphanumeric characters and underscores.
> - The name cannot begin with a number.
> - Do not name the table the same name as an event field or lookup table field.
> - On the lookup table homepage, you can also search for the lookup table by name.

5. (Optionally) Select **Append to existing table** if you want to add the rows to an existing lookup table. See "Appending to an Existing Lookup Table" on page 140 for details.

6. (Optionally) Select **Updatable table** if you want to be able to make updates to the table. See "Updating Lookup Tables" on page 141 for details.

> **Note:** System tables and lookup tables created using the `yieldtable` operator cannot be updated.

7. (Optionally) Enter a description of the lookup table in the Description box.

   (Optional) In the **Tags** field:

   a. Enter or select one or more resource tags to apply to the lookup table. The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.

   b. To create a new tag, type a name containing up to 50 characters and click **Customize tag color** to select a color. See "Creating a Resource Tag" on page 150 to learn all the ways in which you can create resource tags. A new tag is added to the shared tag list immediately, even if you abandon the tag

creation.

c. To apply an existing tag, type in the field and select it from the auto-complete suggestion results.

d. Click **Manage Resource Tags** to view existing or newly created tags.

8. Use the file controls to browse and select your CSV file.

9. Configure "User Sharing and Permissions" on page 130.

> **Note:** Newly created tags default to **Everyone in my Organization (Read)**. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.

10. Click **Create**.

The lookup table is added to the list of lookup tables on the Lookup Tables homepage.

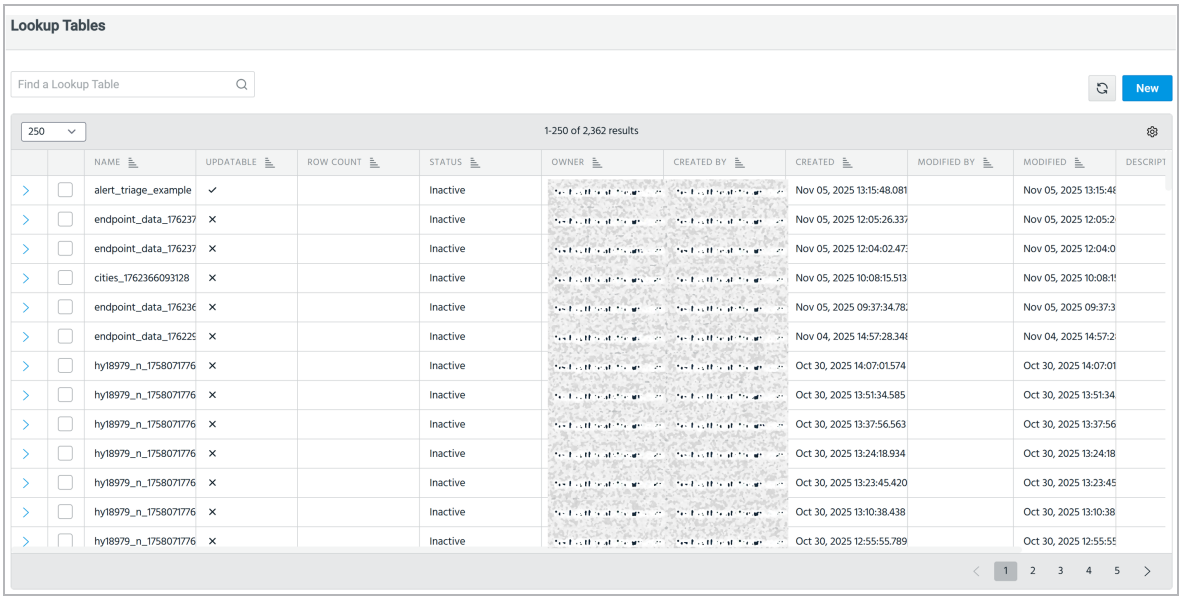
Managing Custom Lookup Tables

From the **Search** > **Manage** > **Lookup Tables** page, you can perform the following actions with custom lookup tables:

- Create a lookup table
- Append to an existing lookup table
- View lookup tables
- Update a lookup table
- Export a lookup table
- Delete a lookup table

Lookup table validation

If there is a problem with the table validation, the status of the table turns to Failed. To know the reason the lookup table activation failed, click **View Details** next to the Failed status.



Appending to an Existing Lookup Table

> **Note:** This feature only enables you to add rows in bulk to an existing lookup table. To delete rows in an existing non-updatable

> lookup table, you must delete the table and upload a new table with the rows you need. To add, edit, or delete rows in an updatable lookup table, use the update feature ( ) on the Lookup Table tab. Refer to "Updating Lookup Tables" below for details.

To append (add rows in bulk) to an existing lookup table:

> **Note:** A user can always only view and edit tables they own. See "User Sharing and Permissions" on page 130 for more information.

1. Create a CSV file that contains the rows you want to add to an existing lookup table. Note that the rows you add must have the same number of columns as the lookup table to which you add the rows.
2. Navigate to **Manage** > **Lookup Tables**.
3. Click the **New**. The New Lookup Table dialog box opens.
4. Click **Append to existing table**.
5. Select the lookup table to which you want to add rows.
6. (Optionally) Enter a description of the lookup table in the Description box.
7. Use the file controls to browse and select your CSV file.
8. Configure "User Sharing and Permissions" on page 130.
9. Click **Create**.

The rows are added to the existing lookup table.

> **Note:** dfgdfgdfgjglhkjfdlhkgjfdlhjfdlhkgk

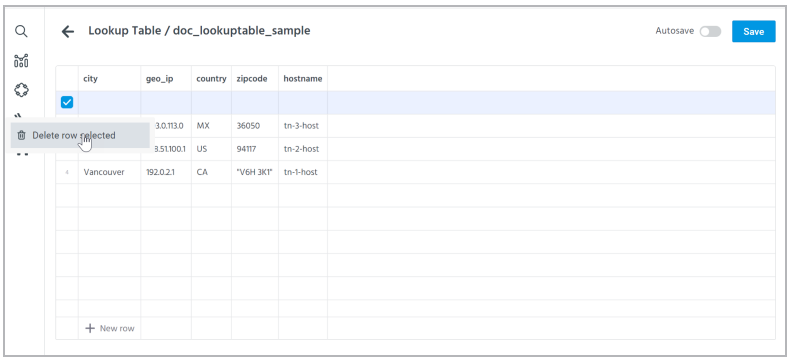
Updating Lookup Tables

> **Note:** A user can always only view and edit tables they own. See "User Sharing and Permissions" on page 130 for more information.

Updatable lookup tables can be modified. You can edit cell values, add rows, and delete rows.

To update a lookup table:

1. Navigate to **Manage** > **Lookup Tables**.
2. Hover your mouse over the lookup table you want to update and click the Edit icon ( ). You are redirected to the table in the edit mode.



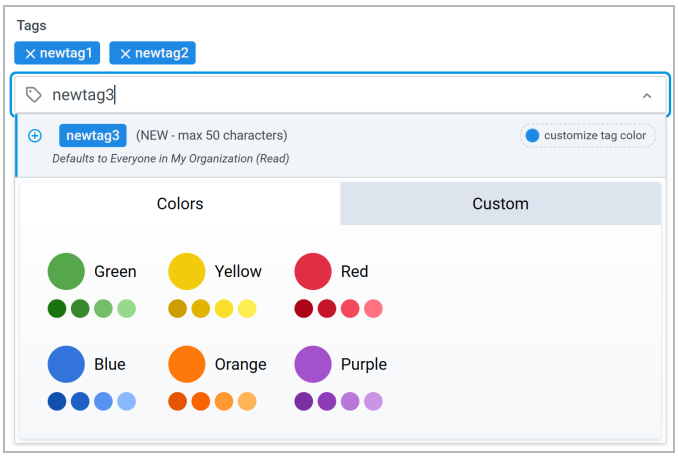
(Optional) In the **Tags** field:

a. Enter or select resource tags to apply to the lookup table. The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.

b. To create a new tag inline, type a name containing up to 50 characters and click **Customize tag color** to select a color. A new tag is added to the shared tag list immediately, even if you abandon updating the lookup table.

c. To apply an existing tag, type in the field and select it from the auto-complete suggestion results.

d. Click **Manage Resource Tags** to view existing or newly created tags.



> **Note:** Newly created tags default to **Everyone in my Organization (Read)**. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.

3. (Optionally) Disable **Autosave** if you do not want to automatically save your updates. When **Autosave** is enabled, updates are auto-saved every minute.
4. Modify the lookup table:
   - To add a new row to the table, click **+ New row**.
   - To delete a row, select the row you want to delete, right click on the selected row, and click **Delete row selected**.
     To delete all rows, select the header row, right click on the rows, and click **Delete rows 1-x**.
   - To edit a cell value, double click on a cell and edit its value. Values in the header row cannot be modified.
5. If **Autosave** is disabled, click **Save**.

The lookup table is updated.

Viewing Lookup Tables

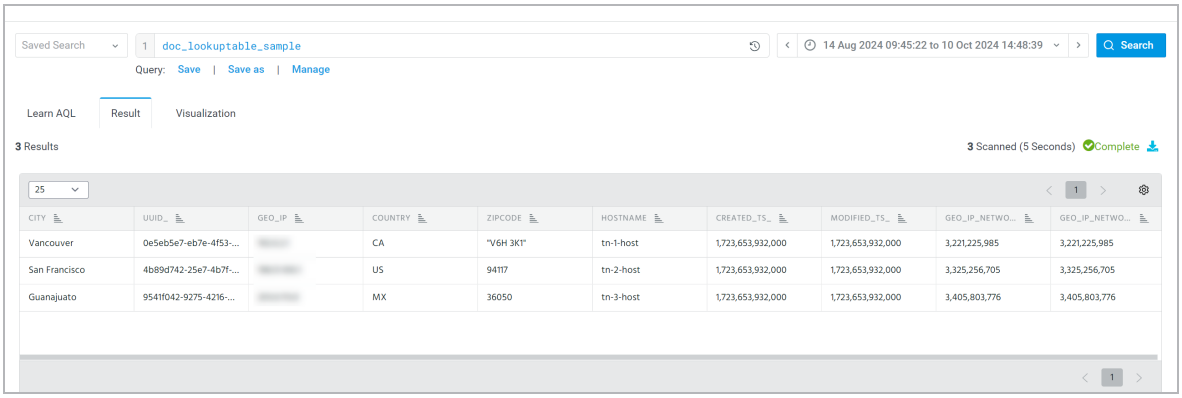
You can view all lookup tables in Search on the Result tab.

> **Note:** A user can always only view and edit tables they own. See "User Sharing and Permissions" on page 130 for more information.

To view a lookup table:

1. Navigate to **Manage** > **Lookup Tables**.
2. Click the active lookup table of interest.

You are redirected to the **Result** tab of the AQL Search page.



Exporting Lookup Tables

Lookup tables can be exported in CSV or JSON format.

To export a lookup table:

1. Navigate to **Manage** > **Lookup Table**.

   Navigate to **Search** > **Manage** > **Lookup Tables**.
2. Hover your mouse over the active lookup table of interest.
3. Click the Export icon ( ).

Alternatively, you can also:

1. Click on the active lookup table name. This opens the Result tab of the AQL Search page.

2. Click the Export icon ( ).
3. Select the format and the number of rows to export.
4. Click **Submit**.

Downloading of the lookup table starts immediately.

Deleting Lookup Tables

To delete a lookup table:

1. Navigate to **Manage** > **Lookup Tables**.
2. Mouse over the lookup table you want to delete and click the Delete icon ( ).

The lookup table is deleted.

Alternately, to delete a lookup table, select a lookup table of your interest and click **Delete**.

Chapter 8: Resource Tags

Anomali users work with many resources such as dashboards, saved searches, alerts, macros, lookup tables, and views. As these resources grow, locating and organizing them across different teams, projects, or use cases could become difficult.

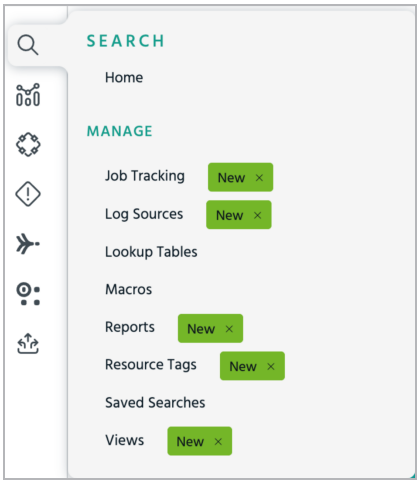
Resource Tags solves this problem by providing a shared, organization-wide tagging system. Any user who can edit a resource can apply tags to it and create new tags.

A single resource can have multiple tags, and one tag can be applied to many resources. This flexibility lets your team organize and find resources as per your needs.

Resource tags are supported for dashboards, saved searches, alerts, macros, lookup tables, views, and reports. See "User Sharing and Permissions" on page 148 to understand who can rename, merge, and delete tags.

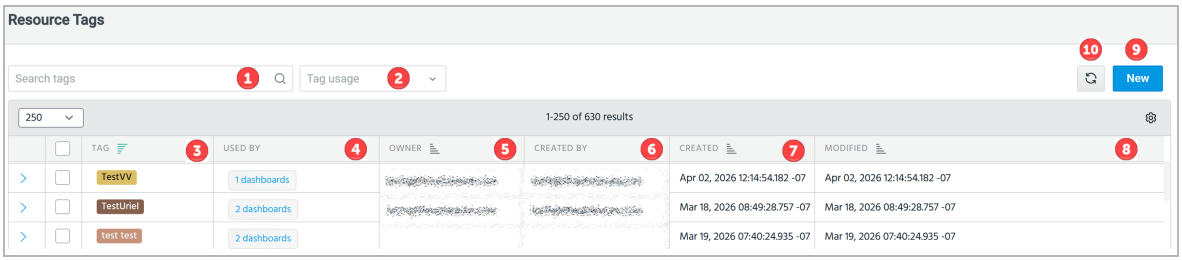
Accessing Resource Tags

To access **Resource Tags**, on the left navigation menu, click **Search** > **Manage** > **Resource Tags**.



Understanding the Tags Catalog

The Resource Tags page is the central catalog for viewing and managing the tag library of your organization. This list of tags is filtered by your permissions. You can view and monitor tags you own or those that have been shared with your organization. Organization administrators see all tags. See "User Sharing and Permissions" on the next page for more information.



1	**Search:** Filter the tag list by name. The search is case-insensitive and updates results incrementally as you type.
2	**Tag usage:** Filter the list by whether tags are currently in use. Possible values are: • **All**: Show all tags visible to you, regardless of use. • **Used**: Show only tags that are applied to at least one dashboard. • **Unused**: Show only tags that have not been applied to any dashboard.
3	**Tag:** View the tag name, displayed as a color chip. • Text color automatically contrasts against the chip color for readability. • If a description is added to the tag, it appears as secondary text below the chip. • A badge on each row indicates the sharing type: **Private**, **Shared (Read)**, or **Shared (Write)**. • All columns are sortable—click a column header to sort, and press `Shift+click` to apply multi-column sorting. Sort and column width preferences are saved per user.

4	**Used By:** Shows the number of resources that currently carry this tag across all supported resource types. • Select a resource type link (for example, a dashboard count) to open that resource's list page, filtered to show only resources tagged with this tag. • The count reflects resources visible to you. Organization administrators see a global count across all users. See "User Sharing and Permissions" below for more information.
5	**Owner:** Shows the user who owns the tag. The owner has full control over the tag, including changing its sharing settings and reassigning ownership. See "User Sharing and Permissions" below for more information.
6	**Created By:** Denotes the email address of the user who created the tag.
7	**Created:** Captures the date and time the tag was created.
8	**Modified:** Captures the date and time the tag was last modified.
9	**Refresh:** Reloads the tag list to reflect the latest state of the catalog.
10	**New:** Opens the **Create Resource Tag** page to let you add a new tag to the catalog. See "Creating a Resource Tag" on page 150 for details.

User Sharing and Permissions

New tags default to **Shared to Organization (Read)**. The tag owner or an org admin can change this at any time.

Permission Levels

- **Private** — Only the tag owner can see and use the tag. It does not appear in the tag catalog or in auto-complete suggestions for any other user, unless they are an organization administrator.

- **Shared to Organization (Read)** — All users can see and apply the tag to resources they have permission to edit, but cannot rename, merge, or delete it. This is the default setting when a new tag is created.
- **Shared to Organization (Write)** — All users can see, apply, rename, recolor, update the description, merge, or delete the tag.

Permissions by Role

Action	Read	Write	Owner	Org Admin
View and apply tag	✓	✓	✓	✓
Rename, recolor, update description	✗	✓	✓	✓
Merge or delete	✗	✓	✓	✓
Change sharing or permission level	✗	✗	✓	✓
Reassign ownership	✗	✗	✓	✓
See private tags owned by others	✗	✗	✗	✓
See global used-by counts	✗	✗	✗	✓

Note: Resource tags do not control access to resources. They are organizational metadata labels only. Applying or removing a tag does not change dashboard visibility, sharing settings, or permissions.

Note: Resource tags are scoped to the organization. A tag created in one customer organization is not visible to any other organization. "Shared" in this context means shared among users within the same organization only.

Applying Tags to Resources

The table above governs what users can do with the *tag itself*. To apply or remove a tag on a resource (such as a lookup table or macro), the user must also have the

appropriate permission on that resource:

- **Read permission on a resource**: The user can see tags applied to the resource but cannot add or remove them.
- **Write permission or ownership on a resource**: The user can apply any tag they can see (based on tag sharing) and can remove tags from the resource.
- **Organization administrators**: Can apply or remove tags on any resource, including those they do not own.

Audit Logging Resource Tags

All changes made to resource tags, including creating, renaming, changing a color or description, merging, and deleting, are recorded in the audit log. Each entry captures the action type, the user who performed it, a timestamp, and the relevant state of the affected tag.

See "Audit Logging for Resource Tags" on page 158 for details on what each operation logs and example AQL queries for retrieving audit log entries.

Creating a Resource Tag

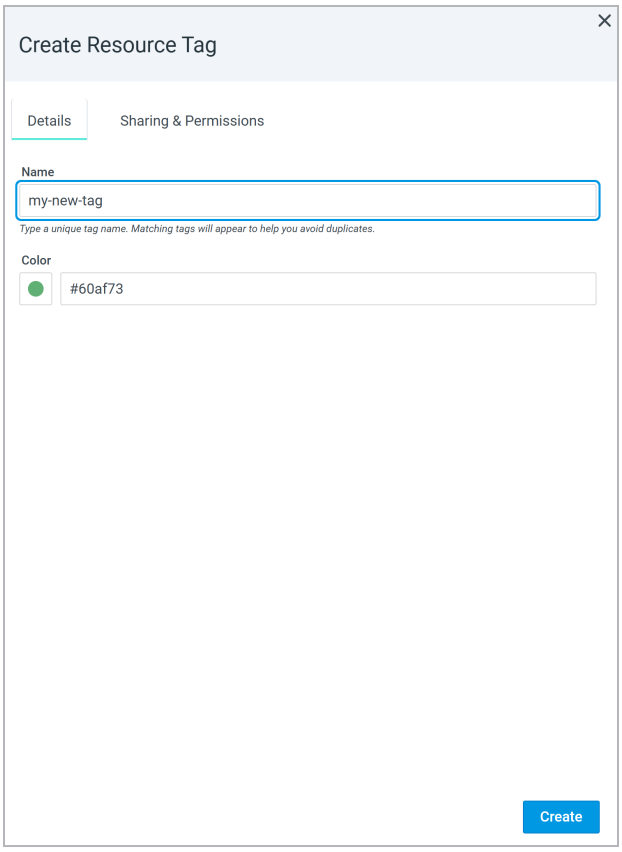
This page describes two ways to create a resource tag:

- Create a tag from the Resource Tags page
- Create a tag inline while editing a resource

Creating a Tag

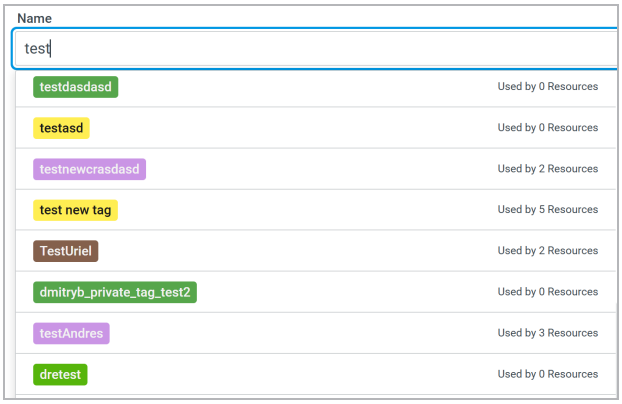
To create a resource tag from the Create Resource Tag page:

1. Click **Search** > **Manage** > **Resource Tags**.
2. Click **New**. This opens the **Create Resource Tag** page on the **Details** tab.



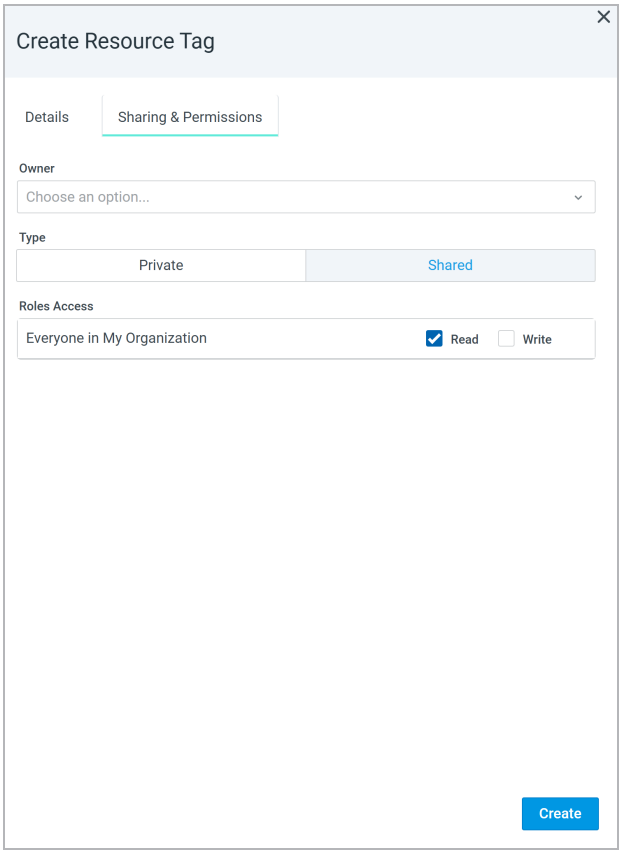
3. In the **Name** field, enter a unique name for the tag, containing a maximum of 50 characters.

   - As you type, matching tags from the catalog appear in a dropdown below the field. This helps you identify near-duplicate tags before creating a new tag.



- If you select an existing tag from the dropdown, the page closes and no new tag is created.
- If the name you enter exactly matches an existing tag (case-insensitive), your submission is rejected.

4. In the **Color** field, select a color for the tag chip. The chip text color automatically contrasts against the selected background for readability.
5. (Optional) Click **Sharing & Permissions** to configure who can see and modify the tag.



- **Owner**: The user who will own the tag. Defaults to the current user. Only the owner or an organization administrator can change sharing settings or reassign ownership after creation.

- **Type**: Select **Private** (visible to owner and organization administrators only) or **Shared** (visible to all users in the organization). Defaults to **Shared**.
- **Roles Access**: If **Shared** is selected, set the permission level for **Everyone in My Organization**:
  - **Read**: Selected by default. Users can view and apply the tag but cannot modify or delete it.
  - **Write**: Users can view, apply, modify, and delete the tag.

See "User Sharing and Permissions" on page 148 for more information on what each permission level allows.

6. Click **Create**. The new tag appears in the **Resource Tags** page immediately, with the correct **Created By** and **Created** values.

Creating a Tag Inline While Editing a Resource

You can also create a tag inline while editing a resource, such as a dashboard, without navigating to the **Resource Tags** page, as follows:

1. Type a new tag name. The creation flow displays auto-complete suggestions as you type.
2. Assign a color to the new tag.
3. Click the add icon (⊕) to add the tag to the dashboard.
4. (Optional) To view or manage existing resource tags, click **Manage Resource Tags**. This launches the **Resource Tags** page containing all the tags in a new tab.
5. Click **Save**.

> **Note:** Inline creation includes **Name** and **Color** only. There is no option to enter a **Description** or configure **Sharing & Permissions**.

Renaming and Merging Resource Tags

You can update the name, color, or description of a tag on the "Resource Tags" on page 146 page. It also supports two distinct workflows depending on what you enter

in the name field:

- **Rename**: Type a new, unique name. The tag is updated in place and all resources, such as dashboards, carrying it reflect the new name immediately.
- **Merge**: Select an existing tag from the auto-complete suggestions. All resources, such as dashboards, currently tagged with the source tag are re-tagged to the target tag, and the source tag is deleted.

> **Note:** You must be the tag owner, have **Write** permission on the tag, or be an organization administrator to rename or merge a tag. You cannot edit tags to which you have **Read**-only access.

> **Note:** A case-only change (for example, renaming *production* to *Production*) is a valid rename action.

Renaming a Tag

To rename a resource tag:

1. Click **Search** > **Manage** > **Resource Tags**.
2. Select the tag you want to rename and click **Edit**. On the **Edit Resource Tag** page:
   - Tag details are pre-populated.
   - Modify the tag name, color, or description.
   - View how many resources, such as dashboards or saved searches, currently use this tag to understand the impact of your update prior to saving it.
3. In the **Name** field:
   - Type the new name.
   - As you type, matching tags from the list appear in a drop-down.
   - If the new name does not match any existing tag, the operation is a standard rename.

- If you select an existing tag from the drop-down, the operation becomes a merge. See "Merging Tags" below to understand how merging different tags works.

4. (Optional) Update the **Color** or **Description** fields.
5. (Optional) Click **Sharing & Permissions** to update the sharing type, permission level, or owner.
   - **Owner**: Reassign tag ownership to another user. Only the current owner or an organization administrator can change these settings.
   - **Type**: Select **Private** or **Shared to Organization**.
   - **Permission level**: If **Shared** is selected, set to **Read** or **Write**.

   See "User Sharing and Permissions" on page 148 for more information.
6. Click **Save**.

The tag name is updated and propagated to all dashboards that carry it.

Merging Tags

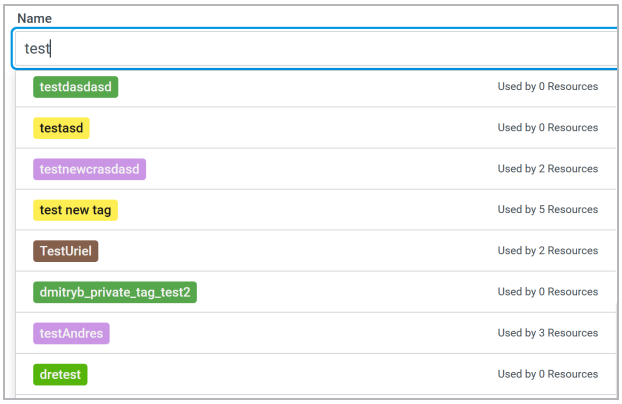
Merging consolidates two tags into one. After a merge, supported resources that carried the source tag are re-tagged to the target tag, and the source tag is permanently deleted from the catalog. The following table shows which resource types support tag merging:

Resource	Supports Tag Merging	Explanation
Anomali Dashboards	Yes	When you merge tags, all dashboards that carried the source tag are automatically re-tagged to the target tag.
Saved Searches	No	Tag associations on saved searches are not updated when you merge tags. Tag changes replace existing values. To update tags on saved searches, edit them individually or use "Bulk Editing Tags on Saved Searches" on page 96.

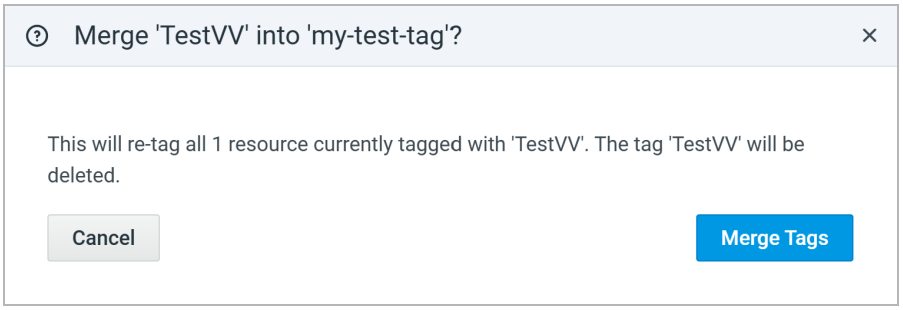
> **Caution:** Merge actions are irreversible, that is, you cannot un-merge tags after a successful merge. Carefully consider merging to avoid misorganized dashboards.

To merge a resource tag into another tag:

1. Click **Search** > **Manage** > **Resource Tags**.
2. Select the tag you want to merge (the source tag) and click **Edit**.
3. In the **Name** field, begin typing the name of the tag you want to merge into (the target tag). Matching tags appear in the auto-complete drop-down.



4. Select the target tag from the drop-down.



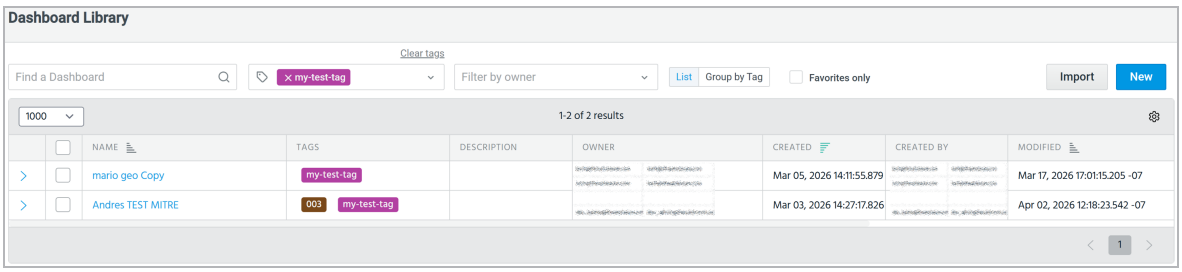
   The dialog shows the source and target tag names, and the number of resources that will be re-tagged.

5. Click **Merge Tags** to confirm the merge operation. After a successful merge:

- All resources previously tagged with the source tag are now tagged with the target tag.
- The source tag is deleted from the catalog.

6. To cancel, click **Cancel**. No change or merging is performed.

After the merge, the **Dashboard Library** reflects the updated tag association. Dashboards that previously carried the source tag now show the target tag, and filtering by the target tag includes all of them.



See Filtering Dashboards by Tag for more information on filtering the dashboard library by tags.

Deleting Resource Tags

You can delete one or more resource tags from the "Resource Tags" on page 146 page. When a tag is deleted, it is removed from the Resource Tags list and untagged from all resources, such as dashboards, that carried it. This action cannot be undone.

> **Note:** You must be the tag owner, have **Write** permission on the tag, or be an organization administrator to delete a tag. You cannot delete tags for which you only have **Read** access.

Deleting a Single Tag

To delete a single resource tag:

1. Click **Search** > **Manage** > **Resource Tags**.
2. Select the tag you want to delete.
3. Click **Delete**. A confirmation dialog displays:

- The tag name.
- The number of resources, such as dashboards, that will be untagged. This count reflects dashboards visible to you. Organization administrators see a global count across all users.

4. Click **Delete** to confirm. The tag is removed from the list and from all associated dashboards.

Deleting Multiple Tags

You can delete multiple tags in a single operation. The **Delete** action is enabled for bulk selection only if you have permission to delete *all* selected tags.

If any selected tag is read-only for your account, the delete action remains disabled.

To delete multiple resource tags:

1. Click **Search** > **Manage** > **Resource Tags**.
2. Select the tags you want to delete. You can select tags across multiple pages.
3. Click **Delete**. A confirmation dialog shows the aggregate impact across all selected tags:
   - The names of all tags to be deleted.
   - The total number of resources that will be untagged. This count reflects resources visible to you. Organization administrators see a global count.
4. Click **Delete** to confirm.

All selected tags are removed from the list and untagged from their associated dashboards.

Audit Logging for Resource Tags

All operations on resource tags such as creating, renaming, changing a color or description, merging, and deleting, write entries to the audit log under the `resourcetag` category. Each entry records the action type, the user who performed it (email and user ID), a timestamp, and the state of the affected tag. The sections below describe what each operation logs and provide example AQL queries for retrieving entries from the audit log.

Tag Creation

Log Details

When a tag is created, the action is logged as an entry in the audit logs and includes:

- `action`: updated with the `created` status
- user details: email and user ID
- timestamp
- other details including name, color, description, initial sharing type, and permission level

Response Digest Updates

The `tag_action` field in the `response_digest` of the audit log is set to `tag.created`.

Accessing the Audit Log

You can access the resource tag audit log entries by launching an AQL Search:

```
audit | where category='resourcetag'
```

Renaming and Merging Tags

Log Details

All rename and merge operations write entries to the audit log. Each logged entry includes:

- `action` field: updated with the `updated` status for both renaming and merge actions
- user details: email and user ID
- timestamp
- the previous and current state of the tag

Response Digest Updates

The audit record is written against the newly merged tag.

Additionally, the `tag_action` field in the `response_digest` of the audit log is updated with one of the following values:

Action	Response Digest Update
Rename	Logged as `tag.renamed`.
Color change	Logged as `tag.color_changed`.
Description change	Logged as `tag.description_changed`.
Merge	Logged as `tag.merged`. Includes the source tag name, target tag name, and the number of resources re-tagged.
Permissions change	Logged as `tag.permissions_changed` when the sharing type or permission level of a tag is updated.

Accessing the Audit Log

You can access the resource tag audit log entries by launching AQL Search queries:

Example 1: Check audit logs for a rename operation

```
audit | where category='resourcetag' and action = "updated"
```

Example 2: Check audit logs for a merge operation

```
audit | where category='resourcetag' and action='updated' and
response_digest contains '"tag_action": "tag.merged"'
```

Example 3: Check actions taken by a specific user

```
audit | where category='resourcetag' and user='john.doe@acme.com'
```

Example 4: Renaming of a specific tag

```
audit | where category='resourcetag' and action='updated' and
response_digest contains '"tag_action": "tag.updated"' and
response_digest contains '"name"'
```

Example 5: Merge where a specific tag was the source

```
audit | where category='resourcetag' and response_digest contains
'"tag_action": "tag.merged"' and response_digest contains
'"source_tag": "OldTagName"'
```

Example 6: Merge where a specific tag was the target

```
audit | where category='resourcetag' and response_digest contains
'"tag_action": "tag.merged"' and response_digest contains
'"target_tag": "SurvivingTagName"'
```

Tag Deletion

Log Details

When a tag is deleted, the action is logged as an entry in the audit logs and includes:

- `action` field: updated with the `deleted` status
- user details: email and user ID
- timestamp
- information about the deleted tag: including its name, color, and description.

Response Digest Updates

For bulk deletes, a separate audit log entry is written for each deleted tag. Additionally, a `tag.deleted` entry is written to the `tag_action` field in the `response_digest` of the audit log.

Accessing the Audit Log

You can access the audit log entries for both single and bulk tag deletion by launching the following AQL query:

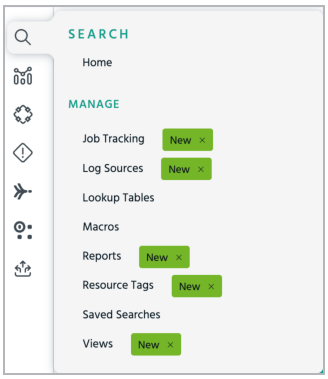
```
audit | where category='resourcetag' and action = "deleted"
```

Chapter 9: Reports

Reports provide a centralized way to turn your operational dashboards into scheduled, repeatable, shareable outputs that can be consumed by external stakeholders outside your SOC or to ensure compliance. Instead of manually exporting data or taking screenshots, you can configure a report once by defining a source dashboard, configuring its variables in the dashboard settings, and setting its schedule.

Accessing Reports

To access reports, click **Search** > **Reports** ( ) in the left navigation panel. The Reports page displays all the information about reports configured to run on a schedule.



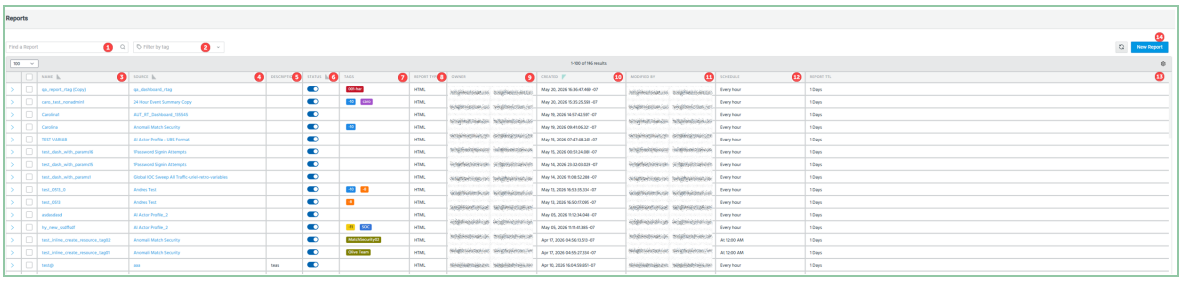
On this page, you can create new scheduled reports, edit existing ones, review run history, and control who can access or modify them by configuring sharing and permissions.

> **Note:** Reports support dashboards built with queries using either the eventlog schema or the OCSF schema, or a combination of both. See "OCSF Schema Overview" on page 39 for more information about the OCSF schema.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature

returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Understanding the Reports Catalog



1	**Find a Report:** Search for an existing report. The search is incremental, that is, the results update as you type, spanning all the columns of the table.
2	**Filter by tag**: Filter the report list by one or more resource tags. • The tag filter drop-down shows only tags you own or that are shared to your organization. Organization administrators can also see private tags owned by other users. • When you select multiple tags, only reports that have all selected tags applied are returned (AND logic). • Click **Clear tags** to remove the filter. • The count next to each tag shows how many reports have that tag applied, not the total across all supported resource types. See "Filtering Reports by Tags" on page 165 for details.
3	**Name:** Name of the report.

4	**Source:** The dashboard source from which this report is generated.
5	**Description:** Optional summary explaining the purpose or contents of the report.
6	**Status:** Indicates whether the report schedule is currently enabled or disabled.
7	**Tags**: Resource tags applied to the report, displayed as colored chips. • Click the add icon (⊕) to add tags to a report. • Click × on a chip to remove a tag from the report. Removing a tag does not delete it from the shared catalog. • The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators. • To create a new tag, type a name and click **Customize tag color** to set a color. New tags default to Shared to Organization (Read). See "Applying Tags to Reports" on the next page for details.
8	**Report Type**: The output formats (HTML or PDF) generated when the report runs.
9	**Owner**: The user who owns the report and controls its configuration.
10	**Created**: When the report was first created.
11	**Modified By**: The last user who changed the report configuration.
12	**Schedule**: How often and when the report is automatically generated.

13	**Report TTL**: How long generated report files are retained before they expire.

Creating Reports

You can create a scheduled report from the Reports page, or directly from any dashboard, which preselects the dashboard as the source and streamlines configuration. See "Creating a Report" below for more information.

Applying Tags to Reports

Resource tags let you label reports with shared, organization-wide identifiers so that you can organize and locate them across your team. Any user with edit access to a report can apply or remove resource tags on it.

See "Applying Tags to Reports" on page 169 for more information.

Filtering Reports by Tags

The Reports page lets you filter the reports catalog by tags and ensure more control to narrow the list to reports that carry one or more specific resource tags.

See "Managing Report Tags" on page 171 for more information.

Bulk Editing Tags on Multiple Reports

You can apply or update resource tags across multiple reports at the same time using the bulk edit option. See "Bulk Editing Tags on Multiple Reports" on page 172 for more information.

Creating a Report

You can create a scheduled report from the Reports page, or directly from any dashboard, which preselects the dashboard as the source and streamlines configuration. For details, see the Creating Scheduled Reports section in Scheduled Reports.

To configure general settings:

1. On the left navigation pane, navigate to **Search** > **Manage** > **Reports**.
2. Click **New Report**.

3. Enter a name for the report.
4. (Optional) Enter a description to provide additional context for other users who may view or manage this report.
5. Use the **Status** toggle switch to enable or disable the schedule.
   a. When on, the report runs automatically according to the configured schedule and destination.
   b. When off, the configuration is saved but no new runs are triggered until you re-enable it.
6. Under Report Configuration:
   a. Select the dashboard you want to export as HTML or PDF.
   b. A dashboard is pre-selected if you navigate here from a dashboard by clicking **Create Scheduled Report**. You can change the dashboard, if needed.

To configure tags:

In the **Tags** field, add or select one or more tags to apply to the report. See "Applying Tags to Reports" on page 169 for details.

To configure report output:

1. Select the report type — HTML or PDF. For dashboard exports, HTML is the default format used to generate static dashboard snapshots. These reports:
   a. Preserve the layout, look, and theme (light or dark) of the dashboard.
   b. Include charts, tables, visual, and text widgets as static content.
   c. Exclude interactive elements such as drill-downs and tooltips.
2. In the Report Destinations section, click **Add** to add one of the following destinations:
   a. Anomali S3: Default storage, accessible for download through the Anomali UI or API. Select the report TTL (1–180 days).
   b. Customer-provided S3 endpoint: Select your S3 as the report destination by providing the S3 bucket name, region, access key, and secret key.

3. Click **Add** to save the report destination you selected. Once configured, the Report Destinations panel lists all destinations where each scheduled run will deliver the generated report. You can add multiple destinations to send the same report to different systems or stakeholders.

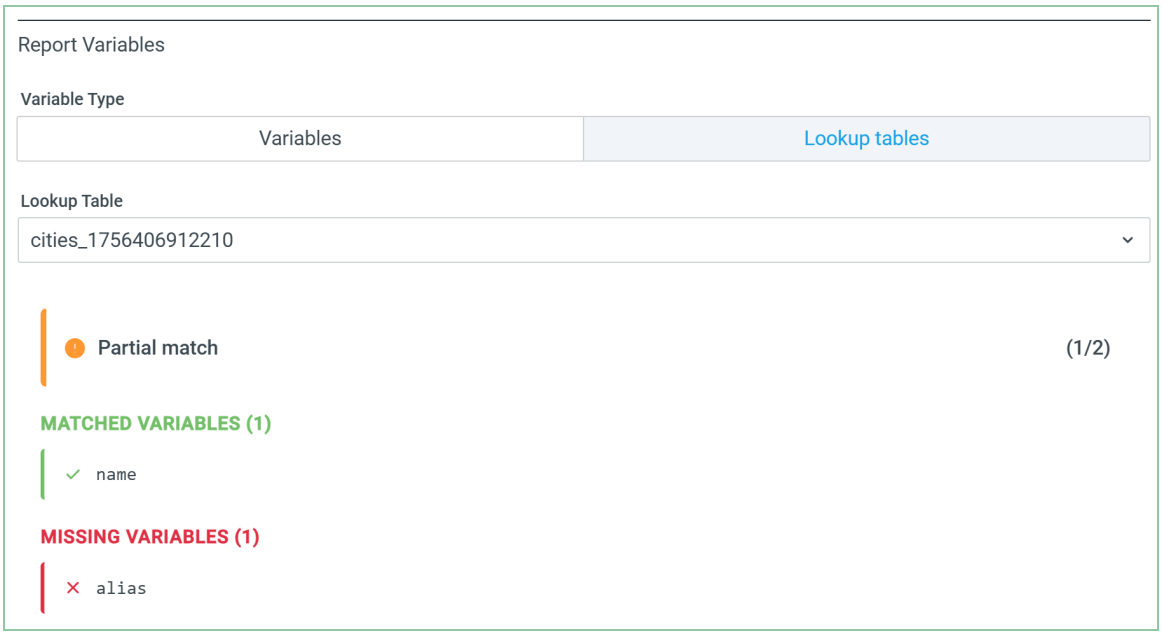
4. Select **Light** or **Dark Mode** as the style theme of the report.

To configure report variables:

Under Report Variables: select either variables or lookup tables to populate the parameter values of the report.

- **Variables**: Select the time range and values for all the other variables present in your source dashboard. These variables are the same variables you configured in your dashboard settings.

- **Lookup Tables**: Select an existing lookup table from a drop-down list to supply values to your source dashboard and automatically generate a report of the dashboard output, without needing any human intervention. To successfully generate a report, all dashboard variables must be present in the lookup table you select. The lookup table supplies up to a maximum of ten rows (one set of values from each table row) for the corresponding dashboard variables.



In the example above:

- A report will not be generated since the lookup table `cities_1756406912210` is missing `alias` as one of its columns, with `name` and `alias` being the two

variables in the source dashboard.

- To correctly generate a report, this lookup table must contain `name` and `alias` as its columns, while supplying up to a maximum of ten rows (one `name` and `alias` from each table row).
- If you configure this report schedule to run at 10 AM with this lookup table that, for example, has three rows, it will run three times at 10 AM, once with each row and produce three output rows.

To configure the schedule:

Specify a scheduling frequency for the report:

- **Standard**: Select a schedule from a list of pre-defined schedules. Possible values include every hour, every four hours, every six hours, every day, or every week.
- **Custom**: Run the schedule after a specified number of hours or days.
- **Cron**: Schedule reports using cron expressions:

  Default: Schedule a cron job to run either every:

  - hour: runs after the specified number of minutes past the hour
  - day: runs every day at the specified time (in 24-hour format)
  - week: runs only on the specified day of the week at the specified time (in 24-hour format)
  - month: runs on the specified day of the month (between 1-31) at the specified time (in 24-hour format)

  Cron: Schedule the cron job using a valid cron expression. For example, using `0 18 14 * *` as your cron expression runs the job at 6:00 PM on day 14 of the month.

To configure sharing and permissions:

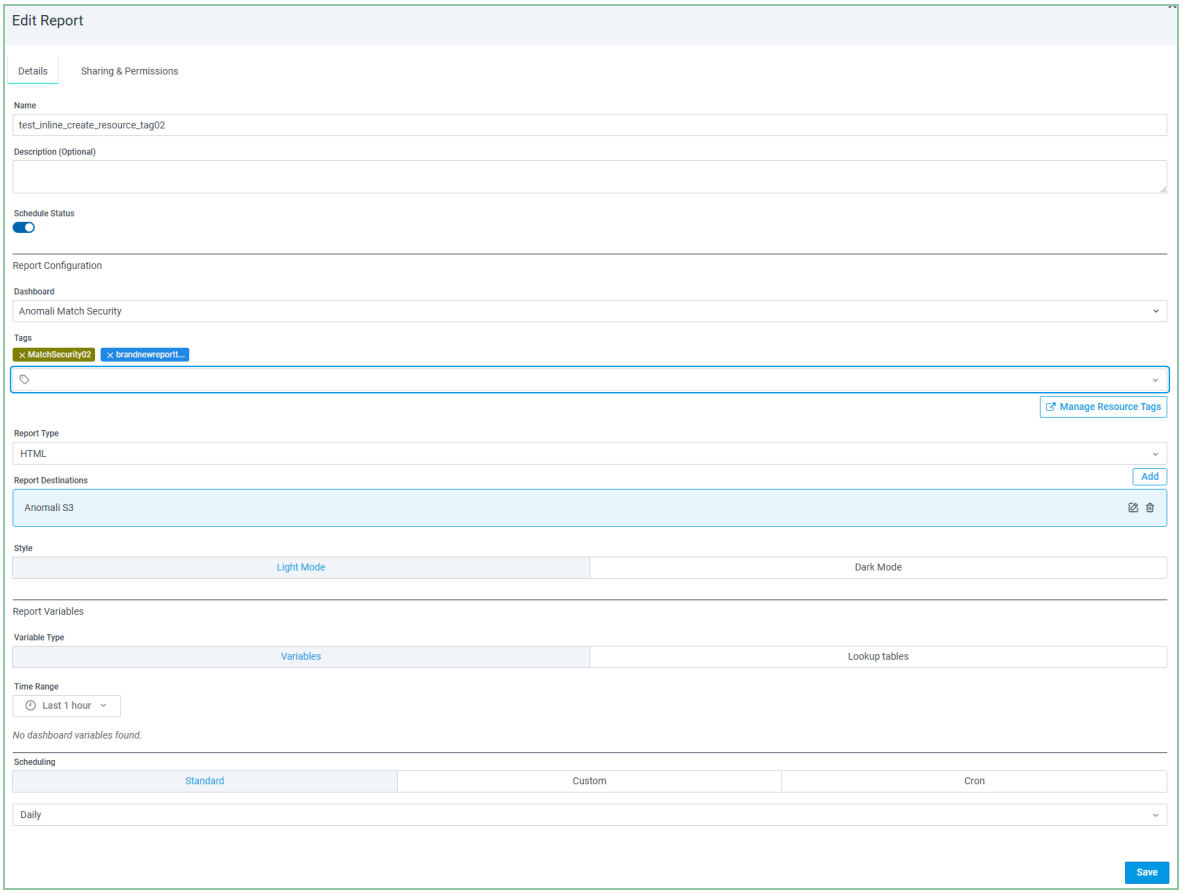
Configure **Sharing & Permissions**:

- Share reports with all org members — newly created reports are readable by everyone in the org by default.

- Assign write-only permission to let users read and modify reports without allowing them to delete them.
- Manage access with admin rights — grant or revoke read or write permissions for other users, keep reports private, or transfer ownership to another user.

Applying Tags to Reports

Resource tags let you label reports with shared, organization-wide identifiers so that you can organize and locate them across your team. Any user with edit access to a report can apply or remove resource tags on it.



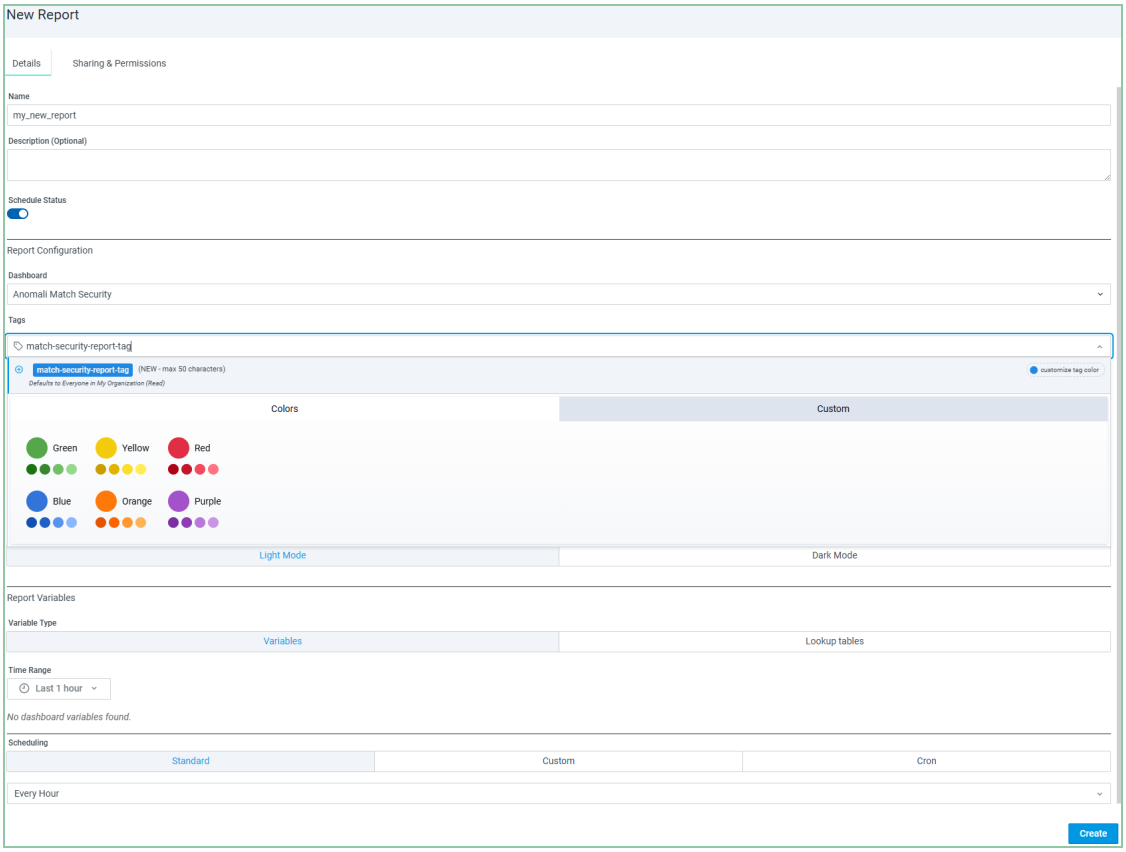
To apply resource tags to a report:

1. On the left navigation pane, navigate to **Search** > **Manage** > **Reports**.
2. Click the name of the report you want to tag.

3. Click the **Tags** field and start typing to search for an existing tag, or type a new tag name.

   - The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.

   - To create a new tag, type a name and click **Customize tag color** to set a color. The color picker lets you choose from preset colors, enter a custom hex value, or select a color from the color spectrum.



4. Select the tag from the drop-down to apply it to the report.

5. Click **Save**.

> **Note:** A new tag is added to the shared catalog immediately, even if you abandon the report edit. The tag association to the report is saved only when you save the report.

> **Note:** Click **Manage Resource Tags** in the Tags field to open the **Resource Tags** settings page where you can view and manage the tags you own or have access to. See "Resource Tags" on page 146 for details.

To remove a resource tag from a report:

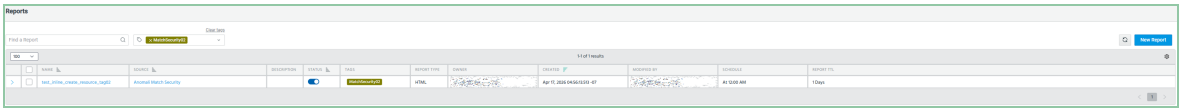
1. Click the edit icon ( ) to open the **Edit Report** page.
2. In the **Tags** field, click × on the tag chip you want to remove.
3. Click **Save**.

Removing a tag from a report does not delete the tag from the shared catalog.

Managing Report Tags

Filtering Reports by Resource Tag

Use the **Filter by tag** control on the Reports page to narrow the list to reports that carry one or more specific resource tags.



To filter reports by resource tag:

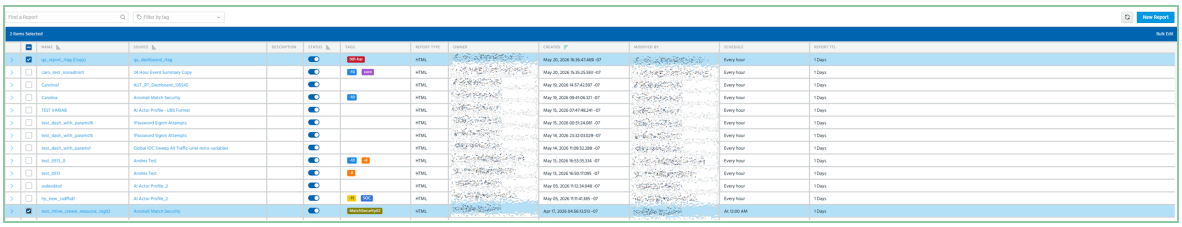
1. On the left navigation pane, navigate to **Search** > **Manage** > **Reports**.
2. Click **Filter by tag** and select one or more tags from the drop-down list.
   - The drop-down shows only tags you own or that are shared to your organization. Organization administrators can also see private tags owned by other users.
   - When you select multiple tags, only reports that have all selected tags applied are returned.
   - The count displayed next to each tag reflects usage for reports only, not the total across all the supported resource types. A tag used across multiple

resource types will show a different count on each resource page, depending on the resource you are viewing.

3. Click **Clear tags** to remove the filter.

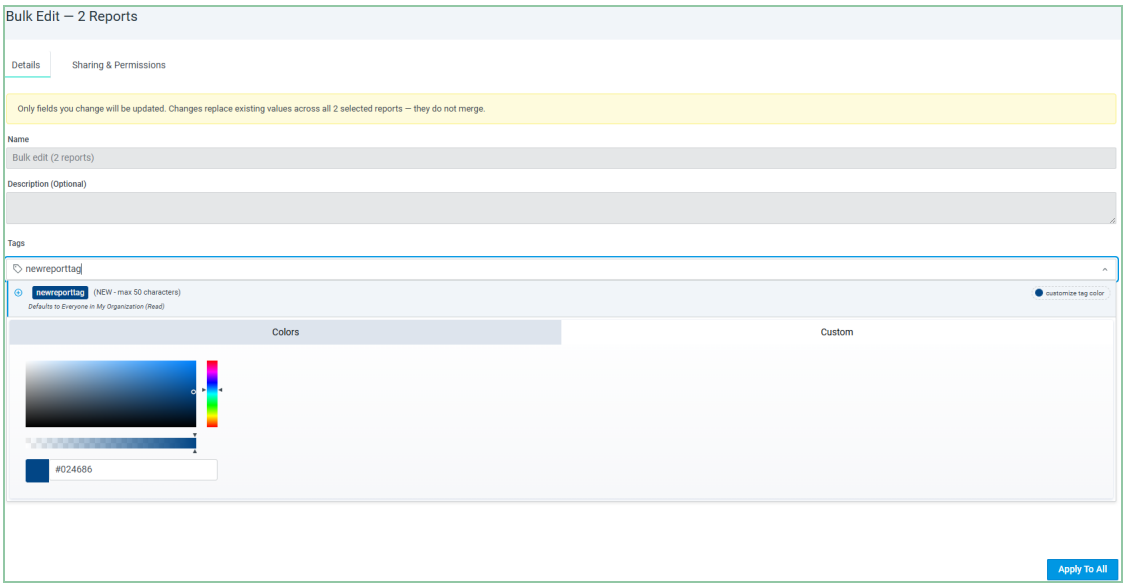
Bulk Editing Tags on Multiple Reports

You can apply or update resource tags across multiple reports at the same time using the bulk edit option. Select two or more reports in the Reports list to open the **Bulk Edit** page.



To bulk edit tags on multiple reports:

1. On the left navigation pane, navigate to **Search** > **Manage** > **Reports**.
2. Select two or more reports.
3. Click **Edit**. This open the **Bulk Edit** page, showing the number of selected reports.
4. On the **Details** tab, click the **Tags** field and select existing tags or type a new tag name to create one. To create a new tag and set its color, type the name and click **Customize tag color**. Click the add icon (⊕) to add any new tags you created.



5. (Optional) Click the **Sharing & Permissions** tab to update the owner or access permissions across the selected reports.

6. Click **Apply to All**.

> **Note:** Only fields you change are updated. Changes replace the existing values across all selected reports — they do not merge with existing values. **Name** and **Description** are read-only in the bulk edit mode.

Exporting and Sharing a Report

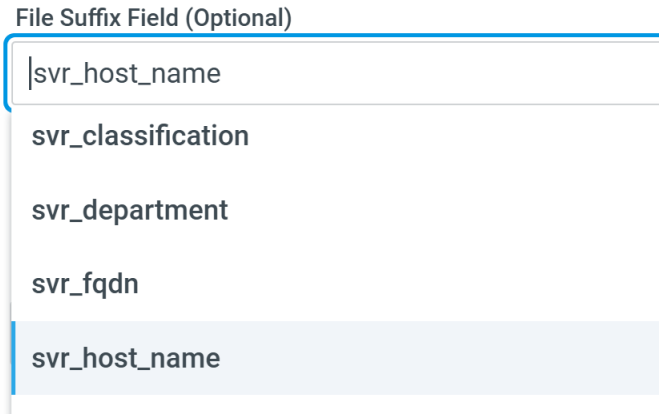
To export or share a report:

1. (Optional) After you successfully create a report, click the share icon ( ) to share and the export icon ( ) to export the report. When you export a report:

   - By default, the file is saved with a filename in the format: `dashboard_<dashboard_name>_<timestamp>_index`, where `<dashboard_name>` is the name of the dashboard associated with the report you want to export, `<timestamp>` denotes the UTC time you exported the report (in `YYYYMMDD_HHMMSS` format), and an `<index>` integer to prevent filename collisions when no suffix is provided.

- Example: `dashboard_Windows_Threat_Hunting_20260422_210857_9`
- In this example, `Windows_Threat_Hunting` is the dashboard with the report, `Apr 22nd 2026 21:08:57 UTC` is the timestamp, and `9` is the index.

- (Optional) You can enter a custom filename by selecting a field from the **File Suffix Field** drop-down list. This list contains columns from the lookup table you selected when creating the report. The filename format becomes: `dashboard_<dashboard_name>_<custom_value>_<timestamp>`, where `<custom_value>` is the value of the selected column.



  - Example: `dashboard_Windows_Threat_Hunting_windows1_20260422_210857`
  - In this example, `Windows_Threat_Hunting` is the associated dashboard, exported on Apr 22, 2026 21:08:57 UTC, followed by the custom suffix `windows1` as the value of the file suffix column `svr_host_name`.

Chapter 10: Views

Search Views allow you to view the default mapping between the event fields of a specific source type and a selected schema, enabling out-of-box content dashboards for a particular sourcetype or OCSF category.

Views support both the Eventlog (standard) schema and the "OCSF Schema Overview" on page 39. With support for creating aliases for fields and directly referencing them in Search queries, views let you perform schema-on-the-fly field extractions for easier searching and data filtering.

Eventlog and OCSF views are subject to their respective eventlog and OCSF data access filters defined in user roles. See "Role-Based Access Control for Schemas" on page 110 for more information.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Benefits

Views offer the following benefits:

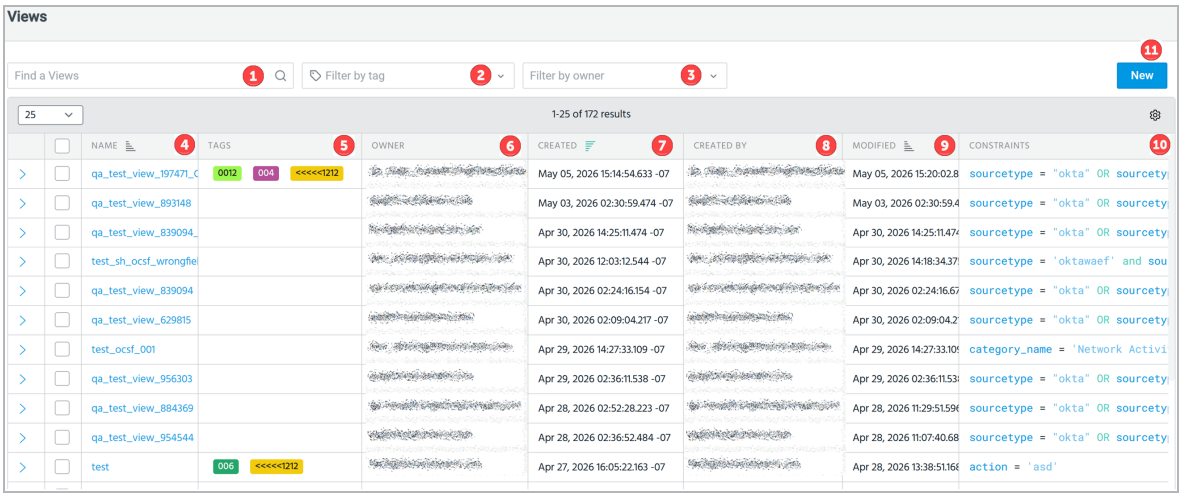
- **Field Recasting**: Recast fields as per the naming convention of a log source you are familiar with, making it easier for you to understand, adapt, and use your data effectively while using Anomali Search.
- **Schema Compatibility**: Use alias fields in search queries as well as filter conditions, as if they were parsed schema fields.
  - Example: The most common AQL syntax is `<alias_field>` = `<value>`.
- **OCSF-Compliant Vendor-Neutral Field Access**: OCSF views bind to an event category rather than a specific log source. A single OCSF view for a category such as Network Activity applies across all OCSF-compliant data sources that produce events in that category, regardless of vendor or format. For example, network events from multiple firewall vendors are all normalized to the same

OCSF Version 1.2 fields, allowing you to apply field aliases present in one OCSF view consistently across all of them. By contrast, eventlog views bind to a specific sourcetype, so each vendor or log format requires its own view.

- **Dynamic Dashboarding**: Create out-of-the-box dynamic dashboards for a sourcetype or OCSF category through a default view.
- **Reduced Data Reprocessing**: Use new fields even against old data, eliminating the need to reprocess old data.
- **Reduced Data Volume**: Leverage the full Common Information Model standard without creating or expanding your existing schema, reducing the data volume.
- **Simplified Usage**: Provide schema-field notation without additional programmatic syntax, simplifying large search queries and improving the overall user experience.
- **MSSP Support**: Views are supported for the Managed Security Service Providers feature.

Overview

To access all views available, navigate to **Search** > **Manage** > **Views**.



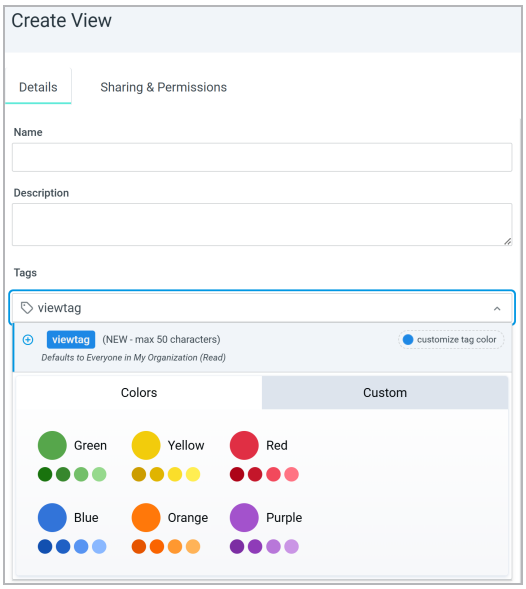
1	**Search View:** Enter the name of the view you want to find. You can also search by name, description, created by, or owner, with the search being incremental, that is, results update as you type.

2	**Filter by tag**: Filter the view list by one or more resource tags. • The drop-down shows only tags you own or that are shared to your organization. Organization administrators can also see private tags owned by other users. • Selecting multiple tags returns only views that have all selected tags applied. • Click **Clear tags** to remove the filter. • The count next to each tag shows how many views have that tag applied, not the total across all the supported resource types. See "Filtering Views by Resource Tag" on page 190 for details.
3	**Filter by owner**: You can filter views by owner. Possible values include: • **Mine**: returns views for which you are the owner. • **System**: returns out-of-the-box views, that is, views that are owned by nobody and that cannot be changed. • **Not System**: returns views that are not owned by system, such as views created by a user. • **<user>**: returns views where the selected user is the owner.
4	**Name:** Name of the view.

5

Tags: Apply one or more resource tags to the view.

- The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.
- To apply an existing tag, type in the field and select it from the auto-complete suggestion results.
- To create a new tag, type a name that does not match any existing tag and confirm the selection. A color picker appears so you can choose a tag color before the tag is created.



- After selecting the color, click the add icon (⊕) to add the tag to the view as well as the shared tag catalog. A new tag is added to the shared tag list immediately, even if you abandon creating or editing the view.

- New tags default to **Everyone in My Organization (Read)**.

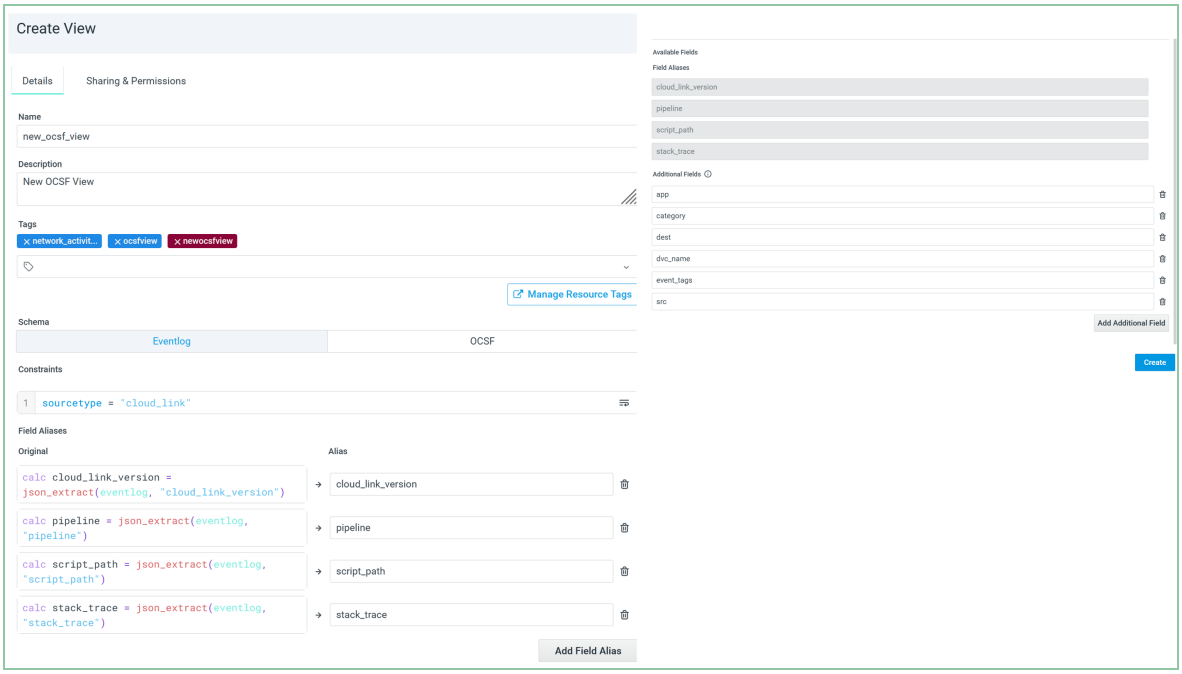
	You can change this later from the "Resource Tags" on page 146 settings page • Applied tags render as colored chips. Click the cross icon (×) on the tag chip to remove the tag from the view. This does not delete the tag from the catalog. Any user with edit access to the view and visibility to a tag can remove the tag. • To filter the list by tag, see "Filtering Views by Resource Tag" on page 190. **Note:** Users with read-only permission on a view can view applied tags but cannot add or remove them. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.
6	**Owner:** The user or role that currently owns and manages the view.
7	**Created:** The date and time when the view was initially created.
8	**Created by**: Indicates the owner of the view. By default, the creating user is the owner and is private (not shared). The owner can: • Modify all the details of the view. • Configure sharing or permissions granted to users. • Edit can delete the view.
9	**Modified:** The date and time when the view was last updated.
10	**Constraints**: The constraint expression that allows you to specify one or more fields to create the view. Constraint fields depend on the schema selected: • eventlog views use `sourcetype` • OCSF views use `category_name`.

11	**Description:** A brief explanation of the purpose or functionality of the view.
12	**New:** Create a new view. Refer to "Create a View" below for details.

Create a View

To create a view:

1. On the left navigation pane, navigate to **Search** > **Manage** > **Views**.
2. Click **New** to create a view.



Name: Required. Enter a unique name for the view. No two views can have the same name.
(Optional) **Description**: Enter a description that describes the view.

(Optional) **Tags**: Enter or select one or more resource tags to apply to the view. • Use the auto-complete suggestion to add a tag to the view. Only tags you own and those shared to your organization are displayed. Private tags associated with other users are not shown, except to organization administrators. • To create a new tag, type a name containing up to 50 characters and click **Customize tag color** to select a color. See "Creating a Resource Tag" on page 150 to learn all the ways in which you can create resource tags. A new tag is added to the shared tag list immediately, even if you abandon the tag creation. • To apply an existing tag, type in the field and select it from the auto-complete suggestion results. • Click **Manage Resource Tags** to view existing or newly created tags.
Schema: Select the schema for the view.

Constraints: Required. Provide one or more constraints that apply to the view. The available constraint fields depend on the schema selected.

- **Eventlog**: The view queries the standard Anomali eventlog schema. Constraints use `sourcetype`-based fields.
  - `sourcetype = "match_cloud"` displays results that originate from Security Analytics.
- **OCSF**: The view queries the "OCSF Schema Overview" on page 39. Constraints use `category_name` to filter by OCSF category, and field aliases use OCSF dotted-notation fields.
  - `category_name = "Network Activity"` displays events from the Network Activity OCSF category.

For the list of supported OCSF categories, see "Supported OCSF Categories" on page 41. An error is displayed if you attempt to save a view without a constraint.

Note:

- Each sourcetype loads a default view that includes multiple normalized fields.
- Default views are read-only. You can clone a default view.
- Views are associated with users mapped to an org role. So, the owner of a view in an org can share it with users mapped to an org role for viewing as well as viewing and editing.
- Any user can create, edit, or clone their view.
- If the sourcetype is unknown, you can use `json_extract` within the calc operator on the `message` field to define aliases for the view. See AQL Best Practices for more information on using `json_extract` efficiently.

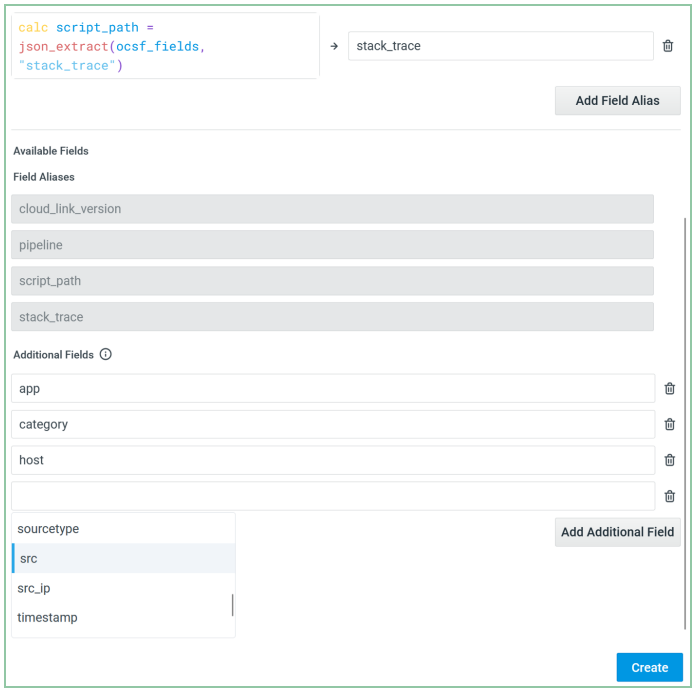
Field Aliases: Required. Enter the name of a field that you would like to alias, the help of auto-complete suggestions. Then, enter its equivalent alias.

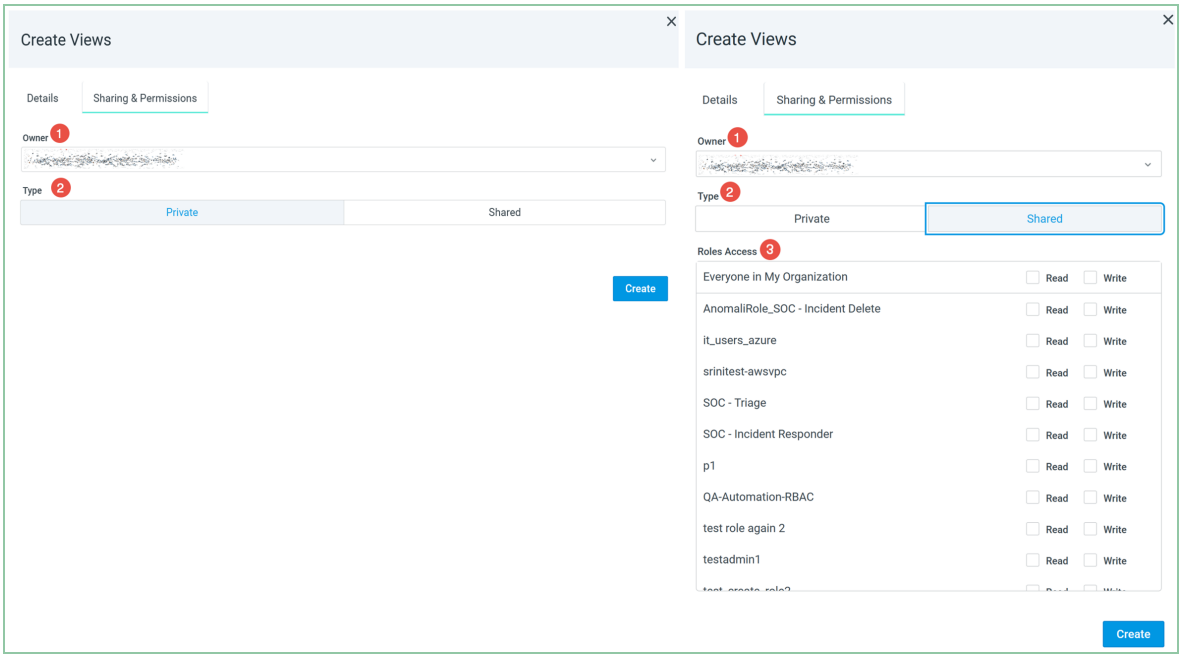
- For **eventlog** views, the fields suggested by the auto-completion depend on the `sourcetype` you provide as the constraint.
- For **OCSF** views, auto-completion suggestions reflect OCSF fields for the selected `category_name`, using dotted-notation (for example, `src_endpoint.ip`, `actor.user.name`). For more information on OCSF field naming, see "Field Naming Conventions" on page 43.
- Aliases should always be a single-word reference, allowing underscores only, without any space in it. The following examples show the correct format:
  - **Correct format**: `source_host_name`
  - **Wrong format**: `"source host name", "source-host-name", or "source.host.name"`.
- You can also add or delete one or more fields and their associated aliases.
- You can enter a field or a query with valid "AQL Syntax" on page 62 to define an alias.

> **Note:** With respect to a view, aliases must be unique. The same alias can be used across multiple views. For example, View A can only have 1 alias `"username"`, but View A and View B can have the same alias `"username"`.

Available Fields: The list of field aliases you added, that will be part of the result set when you are "Running a View" on page 185.

(Optional) **Additional Fields**: Based on the sourcetype you select, you can add any other existing standard eventlog field from a dropdown list, besides the Available Fields, without creating aliases.

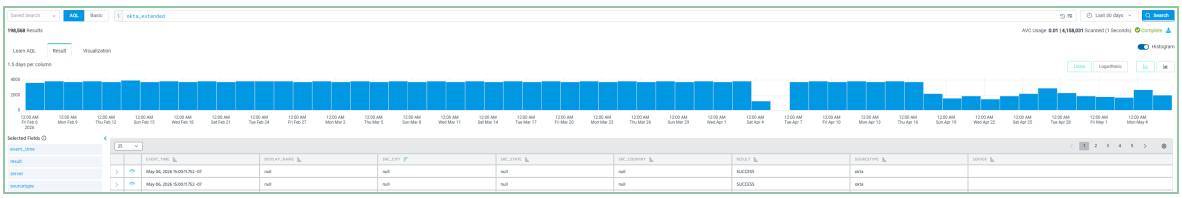




1	**Owner**: Select an owner of the view from the list of users. This is the email address of the owner.
2 3	**Type**: Select the type of permission assigned to the view: • **Private**: Only the owner has read and write privileges, that is, no other user can create, edit, or delete a view. • **Shared**: Allows you to set the read-write privileges for a role that is mapped to different users, granting read-write permissions to all users assigned that role. You can also assign read or write permissions to everyone in your organization. **Note:** Newly created tags default to **Everyone in my Organization (Read)**. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.

Running a View

Once you "Create a View" on page 180, click the view to run it. Once the view runs, it displays all the different fields as defined in the view.



In the example screenshot above:

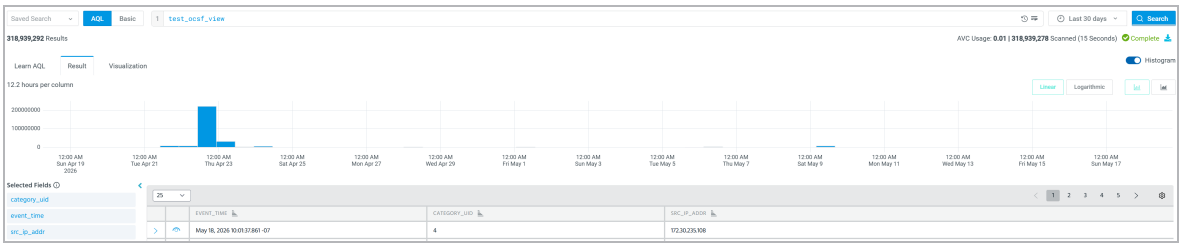
- Results of running the `okta_extended` view contains the user-defined aliases `result`, `server`, `src_city`, and `src_country`, in addition to other fields.
- **Available Fields**:
  - Map to **Selected Fields** in the left pane.
  - Allow you to perform all AQL actions, just like you can on any regular `eventlog` fields.

- The left pane also contains **Additional Fields** that you added when you created the view. For example, `event_time`, `sourcetype`, and `src_ip` are in the **Selected Fields** section because they were added as **Additional Fields** during the view creation.

Additionally, you can run standard AQL queries on the alias fields.

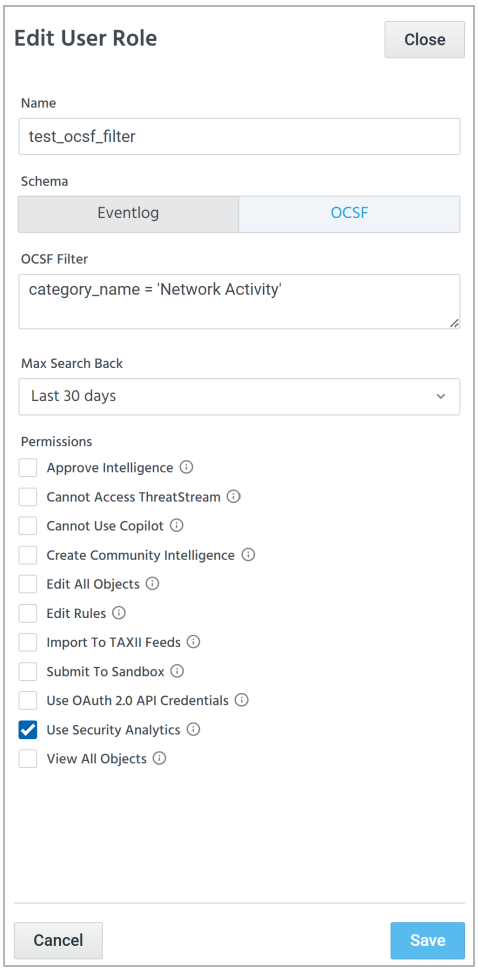
Running an OCSF View

OCSF views run against the OCSF table and return results using OCSF schema fields. The **Selected Fields** pane on the left reflects the OCSF field aliases you defined when creating the view.



In the example above, the `test_ocsf_view` view is constrained to `category_name = "Network Activity"` and aliases the OCSF field `src_endpoint.ip` to `src_ip_addr`. Running this view returns results with the alias field `src_ip_addr` alongside standard OCSF fields such as `category_uid` and `event_time` in the **Selected Fields** in the left pane.

When a non-administrator user with an OCSF role runs an OCSF view, Anomali Search automatically appends the **OCSF Filter** filter to the AQL query of the view in the backend.



The user does not need to add the filter manually, that is, the view returns only the OCSF categories the role permits.

For example, if the role filter is defined as follows

```
category_name = 'Network Activity'
```

Search internally runs the view as if `| where category_name = 'Network Activity'` were part of the view constraint, in addition to the constraint the user specifies. See "Role-Based Access Control for Schemas" on page 110 for more information.

If a user's role grants access to OCSF data but not eventlog data, running a view built on the eventlog schema returns a forbidden error. The underlying AQL for an eventlog view queries the eventlog table, which the OCSF-only role does not permit. Users who need to run both eventlog and OCSF views must have appropriate filters configured for each schema in their assigned roles.

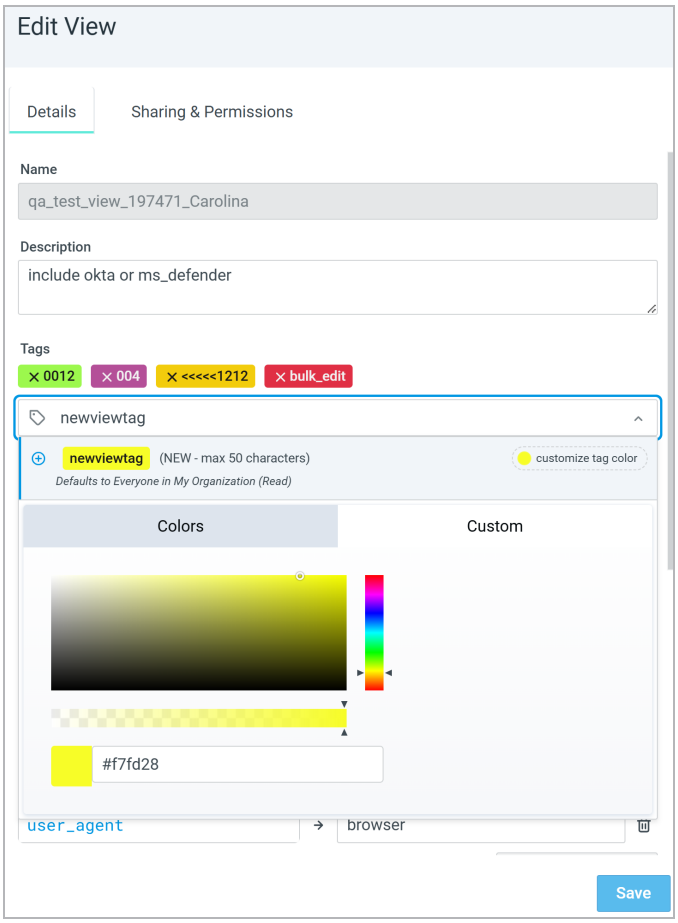
Update a View

To update a view:

1. On the left navigation pane, navigate to **Search** > **Manage** > **Views**.
2. Click on the name of the view to make changes either on the:
   a. **Details** tab: Edit the name, description, constraints, field aliases, available fields, and tags.
   b. **Sharing & Permissions** tab: Edit the owner and the read-write permissions associated with a role mapped to different users.

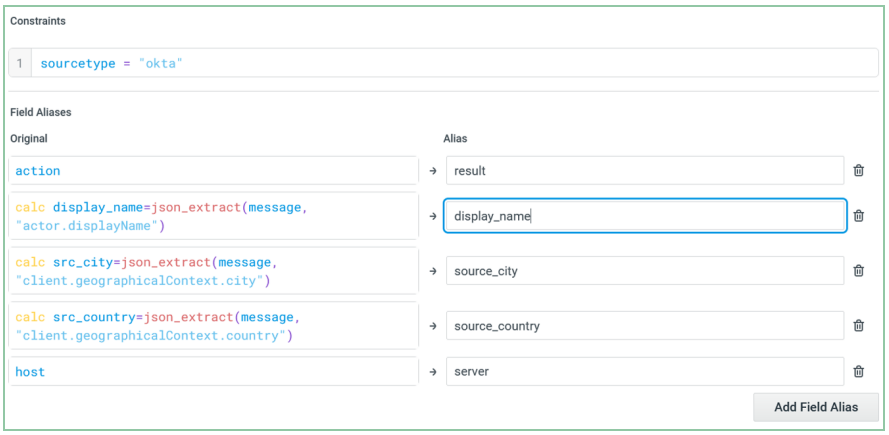
   > **Note:**
   > - When you edit an existing OCSF view, the **Schema** field pre-selects OCSF automatically.
   > - The schema of an existing view cannot be changed after the view is created.

3. (Optional) In the **Tags** field:
   a. Enter or select resource tags to apply to the view. The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.
   b. To create a new tag inline, type a name containing up to 50 characters and click **Customize tag color** to select a color. A new tag is added to the shared tag list immediately, even if you abandon updating the view.
   c. To apply an existing tag, type in the field and select it from the auto-complete suggestion results.
   d. Click **Manage Resource Tags** to view existing or newly created tags.



> **Note:** Newly created tags default to **Everyone in my Organization (Read)**. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.

3. Click **Save**.



In the example screenshot above, the Field Aliases section shows how you can update one or more aliases either by using AQL queries or recasting eventlog fields, as follows:

- `result`: recasted from eventlog field `action`.
- `display_name`: defined by the custom AQL query -> `calc display_name=json_extract(message, "actor.displayName")`.
- `source_city`: updated alias defined by a custom AQL query -> `calc src_city=json_extract(message, "client.geographicalContext.city")`.
- `source_country`: updated alias defined by a custom AQL query -> `calc src_country=json_extract(message, "client.geographicalContext.country")`.
- `server`: recasted from the eventlog field `host`.

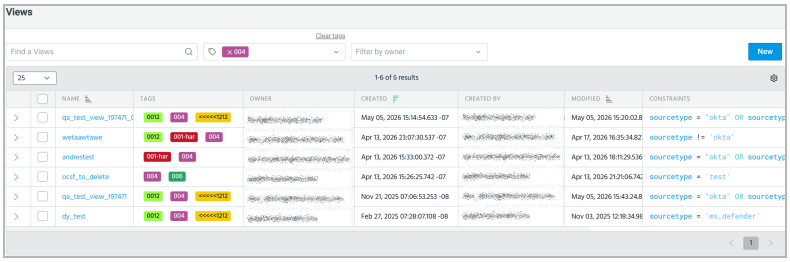
Filtering Views by Resource Tag

You can filter the view list by one or more resource tags. When you select multiple tags, only views that have all selected tags applied are returned.

> **Note:** The tag filter drop-down shows only tags you own or that are shared to your organization. Organization administrators can also see private tags owned by other users. See Resource Tagging Permissions for more information.

To filter views by resource tag:

1. Navigate to **Search** > **Manage** > **Views**.
2. Click the **Filter by tag** drop-down and select one or more tags. The count displayed next to each tag reflects usage for views only, not the total across all the supported resource types. A tag used across multiple resource types will show a different count on each resource page, depending on the resource you are viewing. The list updates to show only views that have all selected tags applied.



3. Click **Clear tags** to clear the filter.

Bulk Editing Tags and Permissions for Views

You can add resource tags and update sharing permissions across multiple views at once.

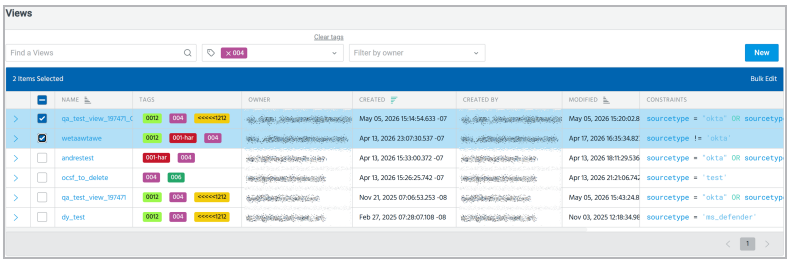
> **Note:** Saving a bulk tag edit replaces the existing tags on all selected views. It does not merge with existing tags.

Required Permissions

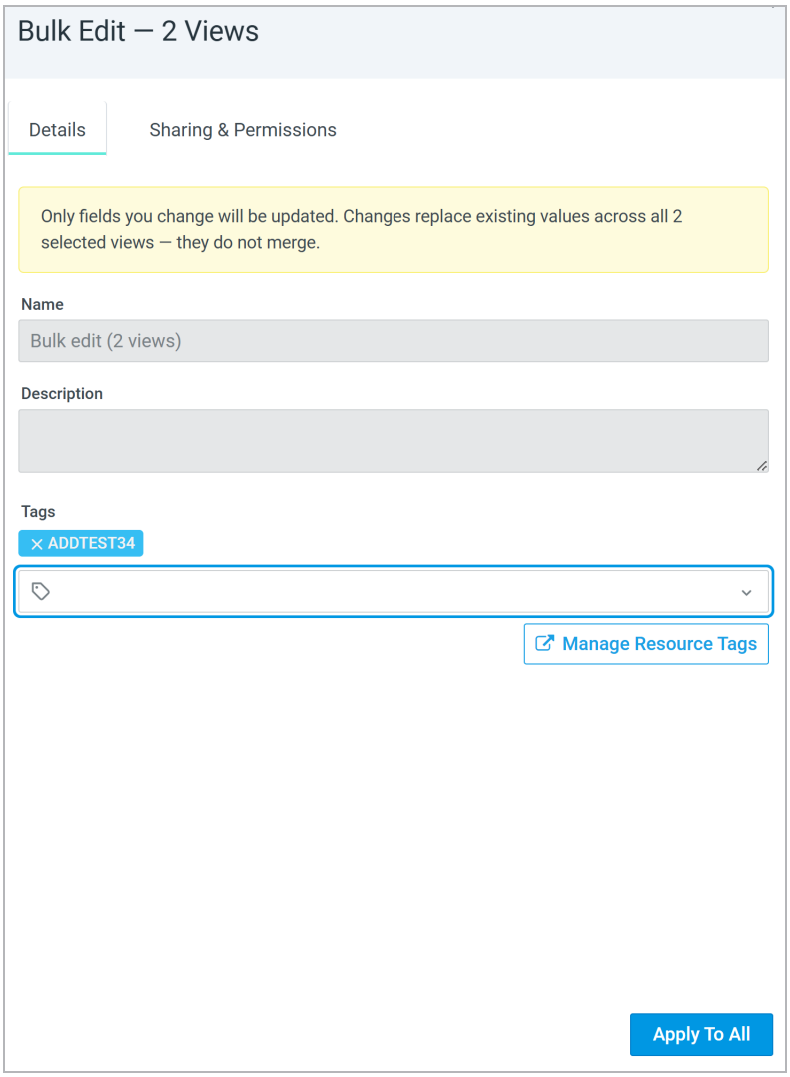
- **Details tab:** Requires write permission or ownership on each selected view. The tag drop-down shows only tags you own or that are shared to your organization. Organization administrators also see private tags owned by other users.
- **Sharing & Permissions tab:** Available to view owners and organization administrators only. Users with write-only permission cannot modify sharing settings.

To bulk edit tags and permissions for views:

1. Navigate to **Search** > **Manage** > **Views**.
2. Select the views you want to update.



3. Click **Bulk edit**.



4. Click the **Details** tab and enter or select tags in the **Tags** field.

5. Enter or select resource tags to apply to the view. The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.

6. To create a new tag inline, type a name containing up to 50 characters and click **Customize tag color** to select a color. A new tag is added to the shared tag list immediately, even if you abandon updating the view.

7. To apply an existing tag, type in the field and select it from the auto-complete suggestion results.

8. (Optional) Click the **Sharing & Permissions** tab to update sharing settings for the selected views. See Resource Tagging Permissions for more information.

9. Click **Apply to All**.

The tags and permissions are updated for all selected views.

Clone a View

To clone a view:

1. On the left navigation pane, navigate to **Search** > **Manage** > **Views**.

2. Hover your mouse over the view you would like to clone. Doing so displays the clone icon, as follows:

3. The cloned view name contains the "`_copy`" suffix added to the name of the original view, as follows:

4. Fill the form to "Create a View" on page 180.

5. Click **Create** to finish cloning.

Delete a View

To delete a view:

1. On the left navigation pane, navigate to **Search** > **Manage** > **Views**.
2. Hover your mouse over the view you would like to clone. Doing so displays the delete icon, as follows:

3. Click the delete icon. Then, click **Delete** to confirm the deletion.
4. The view is now deleted.

Chapter 11: Job Tracking

The turbosearch operator enables you to search for a list of indicators in your data, including historic as well as incoming data.

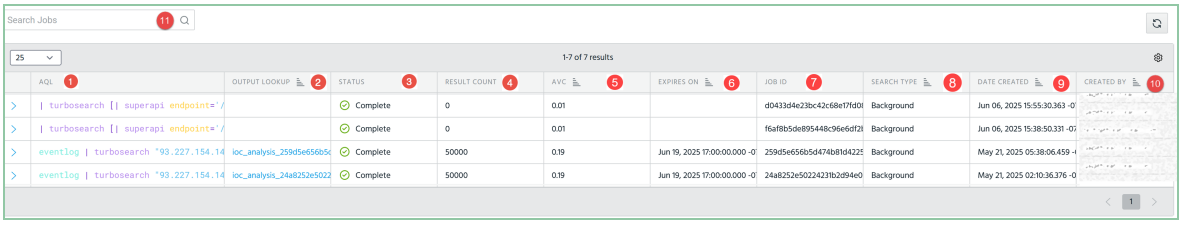
Specifically, you can check whether or not any parsed fields or the raw message contain any indicators of compromise, up to 20 indicators, for the following indicator types:

- IP address
- Filepath
- File names
- File hashes
- Registry keys

If the list of IOCs exceeds 20 indicators, turbosearch will automatically turn this search into a background job.

Monitor background jobs

To monitor a background job, navigate to **Search** > **Manage** > **Job Tracking**.



1	Captures the AQL search expression using turbosearch. See the turbosearch operator page to learn more about creating turbosearch search queries.

2	Output lookup table that is automatically generated by the background job when there are > 20 IOCs. You can click on the generated lookup table to view the results on the Search page for further analysis. If your query uses the `yieldtable` operator, then the output of the background job will be present in the lookup table. See yieldtable for more information. For example, in the query below, the yieldtable operator appends the results into the `myresults` lookup table. `turbosearch [bad_actors | fields ip_addresses] | yieldtable myresults append=true`
3	Indicates the status of the background job. Possible values are: • In Progress: Job is currently running. • Complete: Job is complete, with an automatically generated lookup table as the output. • Canceled: Job was cancelled before completion. No lookup table is generated.
4	Total number of records in the output lookup table the background job automatically generates.
5	Total number of Anomali Virtual Compute (AVC) units the job has consumed. See AVC for more information.
6	Expiry date of the lookup table the background job automatically generates. Default expiry duration is 30 days from the time the job is completed.
7	Job ID of the background job that is automatically triggered by turbosearch, letting you can access the output lookup tables once the job is complete.
8	Type of the job.
9	Date the background job was created.
10	Email credentials of the user that triggered the background job.

11	Search for an output lookup table by its Job ID, once the job is complete.

Chapter 12: Using Macros

This section includes the following topics:

Overview 199
User Sharing and Permissions 202
Creating a Macro 203
Managing Custom Macros 205
Using Custom Macros in Search 208
Viewing Custom Macros 210
Using ThreatStream Macros in Search 210
Available Macros 211

A macro is a reusable, modular piece of an AQL search query that you can insert into other searches. It is designed to help you avoid manually constructing lengthy strings of parameters for complex search queries. You can use pre-built, out-of-the-box ThreatStream macros or create custom macros tailored to your organization's needs. Custom macros created within your organization can be reused in Search, edited, cloned, and deleted.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Key Benefits of Macros

Macros provide the following search benefits:

- **Simplicity**

  You can significantly reduce the complexity of search queries by using macros.

- **Maintainability**

You can automatically update all search queries that include macros by configuring only macro definitions (custom macros only).

- **Consistency**

  You can reduce the amount of search errors due to the same logic used in macros across various queries.

> **Note:** Macros support both the eventlog and OCSF Version 1.2 schema fields with dotted notation. For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`. See "OCSF Schema Overview" on page 39 for more information on the OCSF fields and naming conventions.

For details on creating custom macros, refer to "Creating a Macro" on page 203.

For details on accessing your organization macros, refer to "Viewing Custom Macros" on page 210.

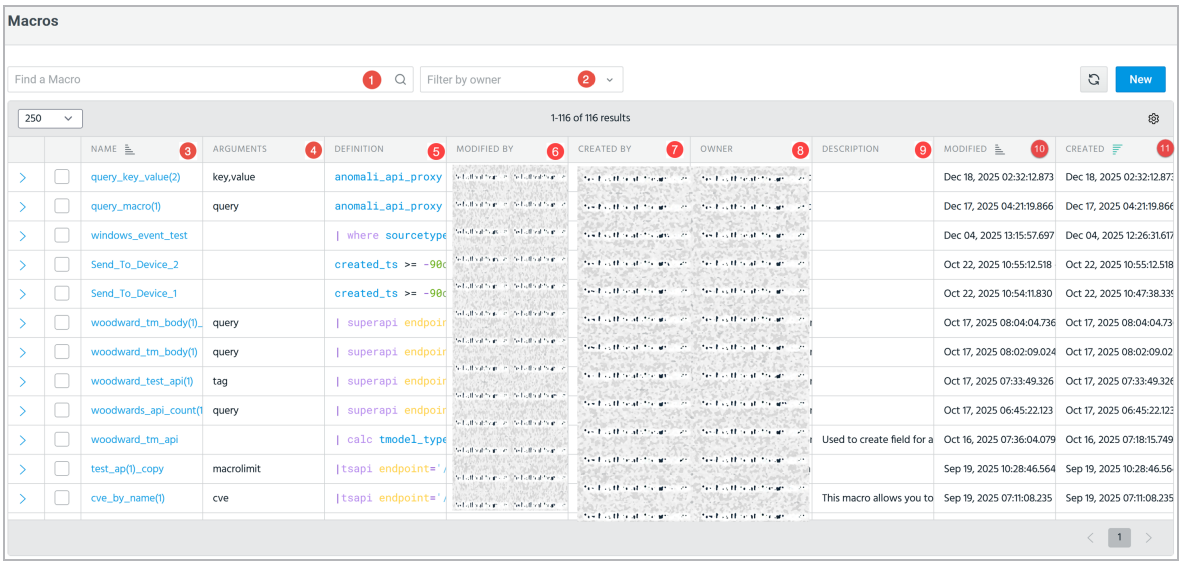
For details on managing custom macros, refer to "Managing Custom Macros" on page 205.

For details on using macros in Search, refer to "Using Custom Macros in Search" on page 208.

For details on using out-of-the-box macros in Search, refer to "Using ThreatStream Macros in Search" on page 210.

Overview

To access all macros available, navigate to **Search** > **Macros**.



1	**Search:** Enter the macro name that you want to find. You can also search by name, description, modified by, created by, or owner, with the search being incremental, that is, results update as you type.
2	**Filter by owner**: You can filter macros by owner. Possible values include: • **Mine**: returns macros for which you are the owner. • **System**: returns out-of-the-box macros, that is, macros that are owned by nobody and that cannot be changed. • **Not System**: returns macros that are not owned by system, such as macros created by a user. • **<user>**: returns macros where the selected user is the owner. **Note:** A user can always only view and edit macros they own. See "User Sharing and Permissions" on page 202 for more information.
3	**Name:** Name of the macro.

4	**Arguments:** The input parameters that the macro accepts when invoked.
5	**Definition:** The AQL query that defines the macro. This definition includes arguments the macro accepts when it is invoked.
6	**Modified by:** The user who most recently updated the macro. **Note:** A user can always only view and edit macros they own. See "User Sharing and Permissions" on the next page for more information.
7	**Created by**: Indicates the owner of the macro. By default, the creating user is the owner and is private (not shared). The owner can: • Modify all macro details. • Configure sharing or permissions granted to users. • Edit can delete the macro. **Note:** A user can always only view and edit macros they own. See "User Sharing and Permissions" on the next page for more information.
8	**Owner:** The user or role that currently owns and manages the macro.
9	**Description:** A brief explanation of the purpose or functionality of the macro.
10	**Modified:** The date and time when the macro was last updated.
11	**Created:** The date and time when the macro was initially created.
12	**New:** Create a new macro. Refer to "Creating a Macro" on page 203 for details.

User Sharing and Permissions

When it comes to macros, the following rules apply to all the different actions users can take:

All Users

- Can always view and edit macros they own.
- See only macros they are entitled to view, based on their role(s) and granted permission(s).
- If no sharing is specified, by default, the macro is private (not shared).
- You can use the "yieldtable" on page 719 operator as it supports ownership and sharing parameters, if you have write permissions.

User Roles with Writer Permissions

Write permission only: you can modify the macro, but not sharing or permission rights.

User Roles with Read Permissions

- Read permission only: cannot edit the macro.

Organization Administrators

- Owner can modify all details and sharing or permission rights for individual users or all users in your org; owner can delete the macros.
- If no owner is specified, by default, owner becomes the user who creates the macro.
- If no sharing is specified, by default, the macro is private (not shared).

Referencing Macros in AQL Search:

- Only visible macros can be referenced.
- Auto-complete feature does not expose macros to which you lack access.
- Structured search also does not expose macros to which you lack access.

Referencing Macros in Copilot:

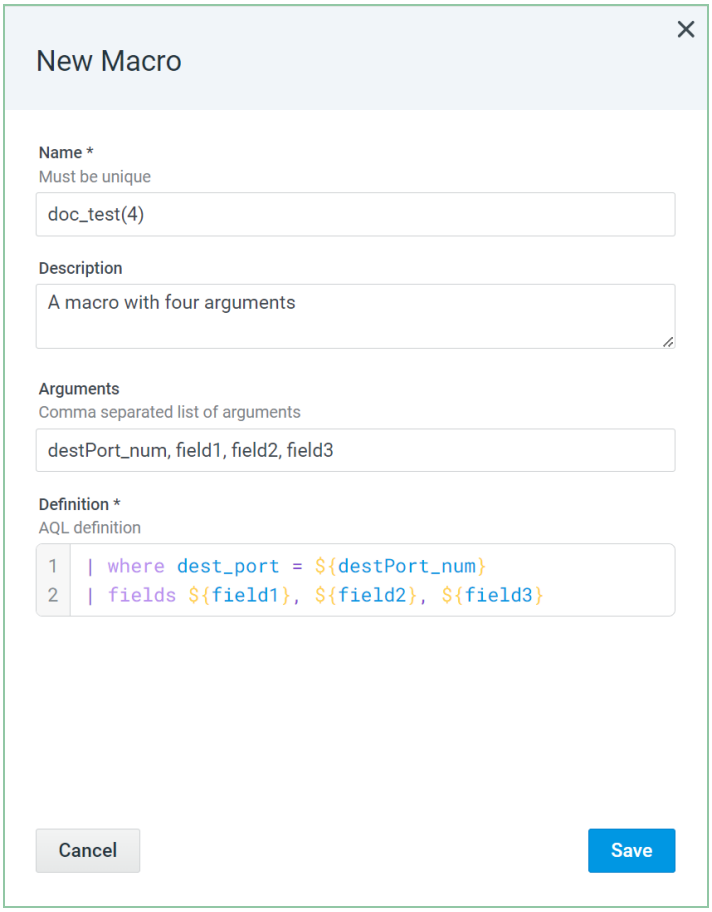
- Generated queries never include macros a user cannot access.

Creating a Macro

On the Macros page, you can create new custom macros that will be available to all users of your organization. Macros can include any part of a search query, such as a `field` list or `where` clause, and do not need to be a complete query. Each macro within the Anomali platform requires a unique name and the AQL definition. The description and arguments are optional.

> **Note:** Macros support both the eventlog and OCSF Version 1.2 schema fields with dotted notation. For OCSF schema fields, array indexes are 1-indexed, that is, they start at `array[1]` and not array[0]. See "OCSF Schema Overview" on page 39 for more information on the OCSF fields and naming conventions.

To create a macro:



1. Navigate to **Manage** > **Macros**.
2. Click **New**.
3. In the New Macro dialog box that opens, provide the following information:
   - **Name:** Enter a unique name for the macro. To be used in Search, only the following characters are accepted in macro names: letters, numbers, underscores (_), hyphens (-), and dots (.). However, the first character of a macro name must be a letter.

     > **Note:** If a macro includes arguments, add the number of arguments in parenthesis after the macro name. For example, `test_macro(3)`, where `(3)` is the number of arguments in the macro.

   - (Optional) **Description:** Enter a description of what the macro can accomplish in Search.
   - (Optional) **Arguments:** Add macro arguments when you need to change or control the output of the macro. Arguments are the actual values that are passed to the macro when it is executed. For example, to search for event traffic going to destination port 443, the search query will be the following: `|where dest_port = 443`. However, using an argument instead of `443` allows any port number to be inserted into the query.
   - (Optional) In the **Tags** field:
     i. Enter or select one or more resource tags to apply to the macro. The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.
     ii. To create a new tag, type a name containing up to 50 characters and click **Customize tag color** to select a color. See "Creating a Resource Tag" on page 150 to learn all the ways in which you can create resource tags. A new tag is added to the shared tag list immediately, even if you abandon the tag creation.
     iii. To apply an existing tag, type in the field and select it from the auto-

complete suggestion results.

iv. Click **Manage Resource Tags** to view existing or newly created tags.

- **Definition:** Enter the AQL search query that will be executed when the macro is inserted in Search. Arguments must be wrapped in `${ }` in the macro definition. Here is an example snippet:

```
| where dest_port = ${destPort_num}

| fields ${field1}, ${field2}, ${field3}
```

4. Click **Save**.

5. Configure "User Sharing and Permissions" on page 202.

> **Note:** Newly created tags default to **Everyone in my Organization (Read)**. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.

The newly created macro is added to the list of custom macros.

To learn how to use macros with arguments in Search, refer to "Using Custom Macros in Search" on page 208.

Managing Custom Macros

Custom macros created within your organization can be edited, cloned, and deleted on the Macros page. These actions can be taken in two ways:

- By selecting a macro and clicking **Edit**/**Clone**/**Delete**.
- By hovering your mouse over a macro and clicking the edit/clone/delete icon.

Editing a Macro

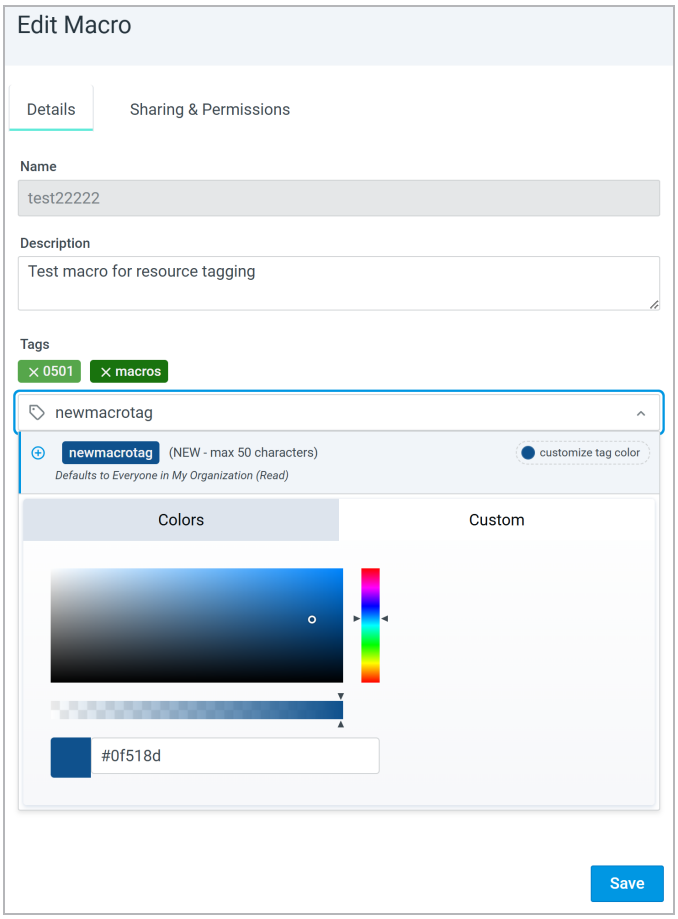
> **Note:** A user can always only view and edit macros they own. See "User Sharing and Permissions" on page 202 for more information.

When editing a macro, you can change its the description, arguments, and definition. The name cannot be modified. To change the name, clone the macro. See "Cloning a Macro" on the next page for details.

Once a macro is changed, the changes are reflected in alerts, saved searches, and dashboards which were created with AQL search queries referencing this macro.

To edit a macro:

1. Navigate to **Manage** > **Macros**.
2. Select the macro you want to edit and click **Edit**. Alternatively, hover the mouse over the macro you want to edit and click .
3. In the Edit Macro dialog box that opens, make the required changes.
4. (Optional) In the **Tags** field:
   a. Enter or select resource tags to apply to the macro. The auto-complete suggestion shows tags you own and tags shared to your organization. Private tags associated with other users are not shown, except to organization administrators.
   b. To create a new tag inline, type a name containing up to 50 characters and click **Customize tag color** to select a color. A new tag is added to the shared tag list immediately, even if you abandon updating the macro.
   c. To apply an existing tag, type in the field and select it from the auto-complete suggestion results.
   d. Click **Manage Resource Tags** to view existing or newly created tags.



5. Click **Save**. The changes are applied to the macros.

Newly created tags default to **Everyone in my Organization (Read)**. Only tag owners and users with write permission can add or remove tags. See Resource Tagging Permissions for more information.

Cloning a Macro

> **Note:** A user can always only view and edit macros they own. See "User Sharing and Permissions" on page 202 for more information.

To clone a macro:

1. Navigate to **Manage** > **Macros**.
2. Select the macro you want to clone and click **Clone**. Alternatively, hover the mouse over the macro you want to clone and click .
3. In the New Macro dialog box that opens, configure the macro name, description, arguments, and definition if necessary.
4. Click **Save**.

The cloned macro is added to the list of macros.

Deleting a Macro

> **Note:** A user can always only view and edit macros they own. See "User Sharing and Permissions" on page 202 for more information.

To delete a macro:

> **Note:** This action impacts all AQL queries and dashboards that use the macro that you are going to delete.

1. Navigate to **Manage** > **Macros**.
2. Select the macro you want to delete and click **Delete**. Alternatively, hover the mouse over the macro you want to delete and click .
3. In the warning message that opens, click **Delete**.

The macro is deleted.

Using Custom Macros in Search

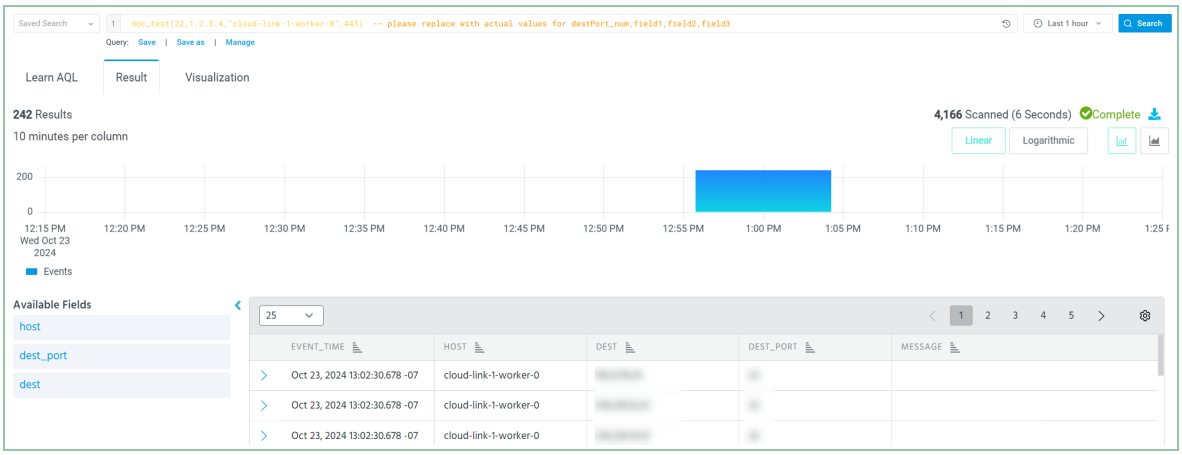
Macros are entered in the Search field with a back tick character (`) before and after the macro name. For example, `` `my_macro(2)` ``. Any macro can be referenced within other macros using the same syntax.

To invoke a macro with arguments, replace arguments with actual values in the order in which they are defined within the macro. For example, in the `doc_test(4)` macro, `destPort_num`, `field1`, `field2`, and `field3` are defined as arguments.

> **Note:** Macros support both the eventlog and OCSF Version 1.2 schema fields with dotted notation. For OCSF schema fields, array indexes are 1-indexed, that is, they start at `array[1]` and not array[0]. See "OCSF Schema Overview" on page 39 for more information on the OCSF fields and naming conventions.

To run the `doc_test(4)` macro in Search, replace the `destPort_num`, `field1`, `field2`, and `field3` arguments with the actual values. The values must be entered in the same order as the text in the arguments. Therefore, the first value is the destination port followed by the fields. For example, `` `doc_test(22,1.2.3.4,"cloud-link-1-worker-0","optic")` ``.

Below is an example of the `doc_test(4)` macro used in Search.



If your argument value is of the string datatype, enclose it in double quotes ("").

In the example above, the argument `evt_msg` takes the string value `Bundle stats` enclosed in double quotes.

Viewing Custom Macros

On the Macros page, you can view all macros created by users of your organization. Macros created within your organization can be referenced in multiple searches without the need to rewrite the entire piece of query.

> **Note:** A user can always only view and edit macros they own. See "User Sharing and Permissions" on page 202 for more information.

To view the list of custom macros, navigate to **Manage** > **Macros**.

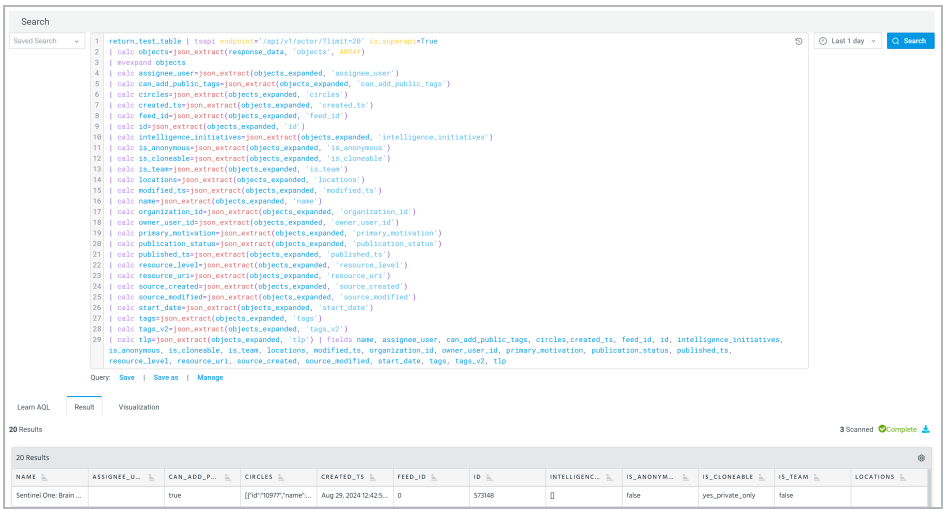
Furthermore, To view a macro definition, expand the macro row of your interest.

On the Macros page, you can take the following actions:

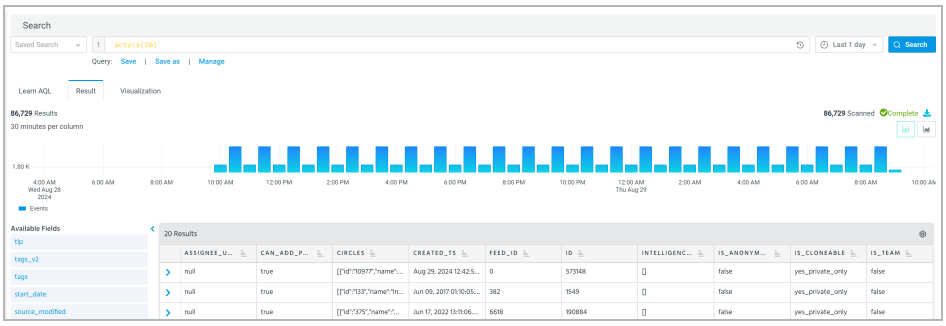
- Create new macros. See "Creating a Macro" on page 203 for details.
- Edit custom macros. See "Editing a Macro" on page 205 for details.
- Clone custom macros. See "Cloning a Macro" on page 207 for details.
- Delete custom macros. See "Deleting a Macro" on page 208 for details.

Using ThreatStream Macros in Search

Anomali offers out-of-the-box macros that enable you to fetch and analyze ThreatStream data without building complex search queries. For example, the image below demonstrates the search query that returns 20 most recent Actors and their field details.

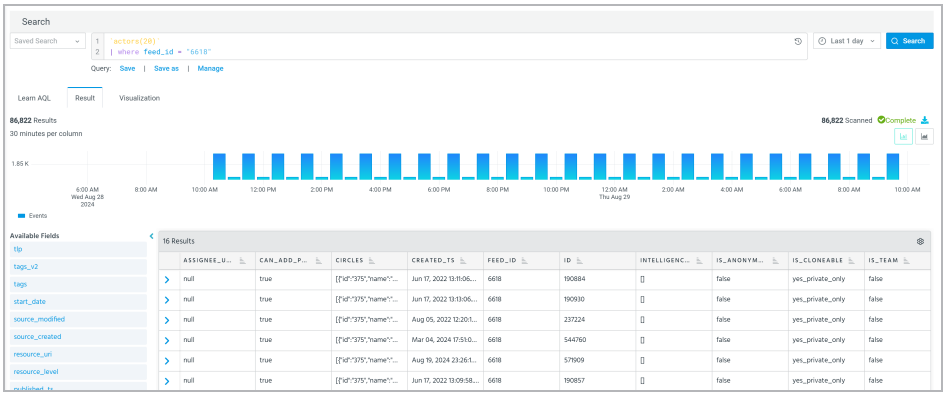


To avoid building such a complex search query, you can use the `actors(20)` macro to return the same results.



Similar to custom macros, out-of-the-box macros are entered in the Search field with a back tick character (`) before and after the macro name. For example, `ioc_feeds`, `malwarethreats(50)`, `rules(30)`.

After running a search query with a macro, you can continue analyzing and filtering data displayed on the Result tab. This can be done by building conditions with available macro fields. See "Available Macros" below for more information about available macros and their fields.



Out-of-the-box macros can be used independently or within a search query.

Search queries containing macros can be saved and managed the same way as regular search queries. See "Anomali Search" on page 21 for details.

Additionally, they can be used to create custom dashboards. See Creating a Dashboard for details.

Available Macros

Use this section as a reference for all out-of-the box macros available in Search. Note that some macros require you to specify a limit for the number of returned

results. The maximum limit you can specify is 1,000. The results are returned based on modified timestamp.

Available macros can be cloned or copied, but they cannot be modified.

actors (limit)

This macro allows you to query the data related to Actors. When used in Search, it returns the fields containing Actor details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	User assigned to this Actor
can-add_ public_tags	Indicates whether the assigned user can add public tags or not. Possible values: true, false.
circles	Trusted circles with which the Actor is shared.
created_ts	UTC time stamp of when the Actor was first created in ThreatStream. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35
feed_id	Numeric ID of the threat feed that provided the threat model entity.
id	ID of the Actor. This ID is automatically assigned to the entity by ThreatStream at the time of creation. You can use this ID to associate the Actor with other threat model entities.
intelligence_ initiatives	Intelligence Initiatives associated with the Actor.
is_anonymous	Indicates whether the Actor user and organization information is anonymized. is_anonymous=1 for true (anonymous) is_anonymous=0 for false (not anonymous)
is_cloneable	Indicates whether the Actor entity can be cloned.

Fields	Description
is_team	Indicates whether the Actor is a team or known to work as an individual (as specified in the Team check box in the UI). True—Actor is a team False—Actor is an individual Default: False
locations	locations applied to the Actor. Possible values include letter codes of regions, countries, and administrative areas (US States and Washington DC only) as defined by STIX 2.1. For example, eastern-africa, central-asia, europe, africa, NL, GB, US-FL, and US-MO.
modified_ts	When the Actor was last updated on the ThreatStream Cloud platform. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35
name	Name of the Actor. Actor name is associated with your organization. Therefore, the name must be unique within your organization. However, two Actors with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that owns the Actor.
owner_user_id	ID of the ThreatStream user that created the Actor.
primary_ motivation	Actor's primary motivation
publication_ status	An Actor can be in the new, pending_review, review_ requested, reviewed, published statuses.

Fields	Description
published_ts	Time stamp of when the entity was published on ThreatStream, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_level	For internal use only.
resource_uri	The API URL to the Actor information.
source_created	Timestamp to when the Actor was first created.
source_ modified	Timestamp to when the Actor was last updated.
start_date	Specify the time when the Actor was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Actor. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Actor.
tlp	Traffic Light Protocol assigned to the Actor. Possible values: red, amber, amber+strict, green, and clear.

attackpattern(limit)

This macro allows you to query the data related to Attack Patterns. When used in Search, it returns the fields containing Attack Pattern details, which then can be included in the operators for further analysis.

Fields	Description
aliases	Other names by which the Attack Pattern is known.
assignee_user	ID of the user to which the Attack Pattern is assigned
can_add_public_tags	Indicates whether a user can add public tags to the Attack Pattern.
circles	All trusted circles to which the Attack Pattern belongs.
created_ts	Time stamp of when the Attack Pattern was created on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Attack Pattern.
id	ID of the Attack Pattern. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Attack Pattern.
is_anonymous	Indicates whether the Attack Pattern user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Attack Pattern can be cloned.
is_mitre	Indicates whether the Attack Pattern is described in the MITRE list. True—if the Attack Pattern is described False—if the Attack Pattern is not described

Fields	Description
is_public	Indicates whether the Attack Pattern is public or private (including belonging to a trusted circle). False—if the Attack Pattern is private or belongs to a trusted circle True—if the Attack Pattern is public
is_team	Indicates whether the Attack Pattern is a team or known to work as an individual. True—Attack Pattern is a team False—Attack Pattern is an individual
killchainstages	List of attack structure phases. Values: `"action-on-objectives", "command-and-control", "delivery", "exploitation", "installation", "reconnaissance", "weaponization"`
locations	Locations applied to the Attack Pattern. Possible values include letter codes of regions, countries, and administrative areas (US States and Washington DC only) as defined by STIX 2.1. For example, eastern-africa, central-asia, europe, africa, NL, GB, US-FL, and US-MO.
modified_ts	Timestamp of when the Attack Pattern was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Attack Pattern. The Attack Pattern name is associated with your organization. Therefore, the name must be unique within your organization. However, two Attack Patterns with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.

Fields	Description
org_id	ID of the organization that created the entity on ThreatStream.
owner_user_id	ID of the user that created the Attack Pattern on ThreatStream.
primary_motivation	Primary motivation of the Attack Pattern.
publication_status	Attack Pattern entities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Timestamp of when the Attack Pattern was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_level	
resource_uri	URI of the Aattack Pattern entity on ThreatStream
source_created	Timestamp to when the Attack Pattern was first created.
source_modified	Timestamp to when the Attack Pattern was last updated.
system	System associated with the Attack Pattern.
start_date	Specify the time when the Attack Pattern was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Attack Pattern. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.

Fields	Description
tags_v2	Additional information on the tags assigned to the Attack Pattern.
target_industry	Target industries associated with the Attack Pattern.
tlp	Traffic Light Protocol assigned to the Attack Pattern. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Attack Pattern for STIX compliance.
workgroups	Workgroups to which the Attack Pattern is visible.

campaign(limit)

This macro allows you to query the data related to Campaigns. When used in Search, it returns the fields containing Campaign details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the Campaign is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Campaign.
circles	All trusted circles to which the Campaign belongs.
created_ts	Time stamp of when the Campaign was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
end_date	The time when this Campaign is known to have ended. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.

Fields	Description
feed_id	Numeric ID of the threat feed that provided the Campaign.
id	ID of the Campaign. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Campaign.
is_anonymous	Indicates whether the Campaign user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Campaign can be cloned.
is_public	Whether the Attack Pattern is public or private (including belonging to a trusted circle). False—if the Attack Pattern is private or belongs to a trusted circle True—if the Attack Pattern is public
modified_ts	Time stamp of when the Campaign was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Campaign. The Campaign name is associated with your organization. Therefore, the name must be unique within your organization. However, two campaigns with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Campaign on ThreatStream.

Fields	Description
owner_user_id	ID of the user that created the Campaign on ThreatStream.
publication_status	Campaign entities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Time stamp of when the Campaign was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Campaign entity on ThreatStream.
source_created	Timestamp to when the Campaign was first created.
source_modified	Timestamp to when the Campaign was last updated.
start_date	Specify the time when the Campaign was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Campaign. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Campaign.
target_industry	Target industries associated with the Campaign.
tlp	Traffic Light Protocol assigned to the Campaign. Possible values: red, amber, amber+strict, green, and clear.

Fields	Description
uuid	UUID (universally unique identifier) assigned to the Campaign for STIX compliance.
workgroups	Workgroups to which the Campaign is visible.

incident(limit)

This macro allows you to query the data related to Incidents. When used in Search, it returns the fields containing Incidents details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the Incident is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Incident.
circles	All trusted circles to which the Incident belongs.
created_ts	Time stamp of when the Incident was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
end_date	The time when this Incident is known to have ended. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Incident.
id	ID of the Incident. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Incident.

Fields	Description
is_anonymous	Indicates whether the Incident user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Incident can be cloned.
is_public	Indicates whether the Incident is public or private (including belonging to a trusted circle). False—if the Incident is private or belongs to a trusted circle True—if the Incident is public
modified_ts	Timestamp of when the Incident was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Incident. The Incident name is associated with your organization. Therefore, the name must be unique within your organization. However, two Incidents with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Incident on ThreatStream.
owner_user_id	ID of the user that created the Incident on ThreatStream.
publication_status	Incident entities can be in new, pending_review, review_requested, reviewed, or published statuses.

Fields	Description
published_ts	Timestamp of when the Incident was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Incident entity on ThreatStream.
source_created	Timestamp to when the Incident was first created.
source_modified	Timestamp to when the Incident was last updated.
start_date	Specify the time when the Incident was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Incident. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Incident.
target_industry	Target industries associated with the Incident.
tlp	Traffic Light Protocol assigned to the Incident. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Incident for STIX compliance.
workgroups	Workgroups to which the Incident is visible.

investigations

This macro allows you to query statistics of the investigations accessible via Anomali platform. When used in Search, it returns the fields containing information about investigations, which then can be included in the operators for further analysis.

Fields	Description
created_ts	Timestamp when the investigation was created.
description	Short description of the investigation.
id	Unique investigation ID. This identifier is assigned to the investigation when it is created on ThreatStream.
modified_ts	Timestamp when the investigation was last modified.
name	Investigation name.
priority	Priority assigned to the investigation. Possible values: very low, low, medium, high, very high
status	Status assigned to the investigation. Possible values: completed, in-progress, pending, unassigned
tags	Tags associated with the investigation.
tlp	TLP setting for the investigation. In the context of investigations, the TLP color sets the maximum level of information that users outside your organization have access to when viewing your investigation.
assignee_id	ID of the user or workgroup to which the investigation is assigned.
assignee_type	Type of assignee: "user" or "tsworkgroup"
elements	Number of entities associated with the investigation.

Fields	Description
r_type	Type of entity associated with the investigation. Possible values listed below. l Actors:actor l Campaigns:campaign l Incidents:incident l Observables:intelligence2 l Signatures:signature l ThreatBulletins:tipreport l TTP:ttp l Vulnerabilities:vulnerability
r_id	Unique ID assigned to the entity.
add_related_ indicators	When enabled, observables related to the entity you are associating with the investigation are also added. add_related_indicators=1 for true (add) add_related_indicators=0 for false (do not add)
tips	For internal use only.
workgroups	Workgroups with which the Investigation is shared.
assignee	User or workgroup to which the investigation is assigned.
circles	Trusted circles with which the investigation is shared.
intelligence_ initiatives	Intelligence Initiatives associated with the Investigation
owner_org	Organization of the owner. In most cases, this is the same as the assignee organization.
owner_org_id	Organization ID of the owner. In most cases, this ID is the same as the assignee organization ID.
resource_uri	Internal URL for the Investigation within ThreatStream.
source_type	Source from which the investigation was created.

ioc(limit)

This macro allows you to query details of observables. When used in Search, it returns the fields containing observable details, which then can be included in the operators for further analysis.

Attribute	Description
asn	Autonomous System (AS) number associated with the indicator.
can_add_ public_tags	Indicates whether a user can add public tags to an observable.
confidence	Level of certainty that an observable is of the reported indicator type. Confidence scores range from 0-100, in increasing order of confidence, and is assigned by ThreatStream.
country	Country associated with theobservable.
created_by	User that created the intelligence.
created_ts	When the observable was first seen on the ThreatStream cloud platform. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35.
description	This attribute is for internal use and can be ignored.
detail2	This attribute is for internal use and can be ignored.
expiration_ts	Time stamp of when intelligence will expire on ThreatStream, in UTC time.
feed_id	Numeric ID of the threat feed that generated the observable. feed_id = 0 for user-created observables.

Attribute	Description
id	Unique ID of the observable. This identifier is assigned to the observable when it is first created on ThreatStream. Unlike update_id, this identifier never changes as long as the observable is available on ThreatStream.
ip	The IP address associated with the observable if the imported indicator is a domain or a URL.
is_anonymous	Whether the organization and user information is anonymized when the observable is accessed by users outside of the owner organization.
is_editable	Indicates whether the imported entity can be updated by an intelligence source. This attribute is reserved for intelligence source providers and can be ignored.
is_public	Visibility of the observable—public or private. 0/False—if the indicator is private or belongs to a Trusted Circle 1/True—if the indicator is public **Default:** 0/False **Note:** This attribute is treated as a string or boolean in requests and as boolean in responses.
itype	Indicator type.
latitude	Latitude associated with the Geo location of the IP.
longitude	Longitude associated with the Geo location of the IP.
meta	Tags associated with the observable. These tags can be used to search for related observables.

Attribute	Description
modified_ts	When the indicator was last updated on the ThreatStream Cloud platform. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35
name	Name of the observable.
org	Registered owner (organization) of the IP address associated with the observable.
owner_ organization_id	ID of the organization that brought in the observable, either through a threat feed or through the import process.
rdns	Domain name (obtained through reverse domain name lookup) associated with the IP address that is associated with the observable.
retina_ confidence	Confidence score assigned to the observable by Anomali machine learning algorithms.
severity	Severity score of the observable.
sort	This attribute is for internal use only and can be ignored.
source	Usually an obfuscated description of the source of the observable; the description may not be obfuscated if the user belongs to your organization. Examples of source names: threat feed name, Analyst (if the user name is obfuscated).
source_ locations	Source locations associated with the indicator. Possible values include letter codes of regions, countries, and administrative areas (US States and Washington DC only) as defined by STIX 2.1. For example, `eastern-africa`, `central-asia`, `europe`, `africa`, `NL`, `GB`, `US-FL`, and `US-MO`.

Attribute	Description
source_ reported_ confidence	A risk score from 0 to 100, provided by the source of the observable.
status	Status assigned to the observable. For example, active, inactive, falsepos.
subtype	For hash observables—those with type=md5—subtype provides additional metadata on the type of hash associated with the observable. Possible values: MD5, SHA1, SHA256, SHA512.
tags	Tags applied to the specified observables.
target_industry	Target industries associated with the observable.
threat_type	Summarized threat type of the observable. For example, malware, compromised, apt, c2, and so on.
tlp	Traffic Light Protocol designation for the intelligence—red, amber, amber+green, clear.
trusted_circle_ ids	IDs of the trusted circles with which the indicator is shared.
type	Observable type—domain, email, ip, md5, string, url.
update_id	An incrementing numeric identifier associated with each update to intelligence on ThreatStream. Every time observable data changes on ThreatStream (for example, if it is updated or expires), the update_id of the indicator is incremented and stored in the observable. This identifier is mostly used by software that interact with ThreatStream to download intelligence, such as ThreatStream Link and ThreatStream APIs.
uuid	UUID (universally unique identifier) assigned to the observable for STIX compliance.

Attribute	Description
value	Value of the observable. For example, value = <192.168.0.10> or value = <http://www.google.com>.
workgroups	Workgroups associated with the observable.

ioc_contributions

This macro allows you to query statistics of the intelligence contributed to by your organization and community. When used in Search, it returns the fields containing information about contributed observables, which then can be included in the operators for further analysis.

Read more...

Fields	Description
community	Total number of observables contributed by the community.
community_ delta	The delta of the number of observables contributed by the community in the specified timespan as compared to the previous timespan. The "-" sign is used to indicate that less observables were contributed than during the previous timespan.
organization	Total number of observables contributed by the your organization.
organization_ delta	The delta of the number of observables contributed by your organization in the specified timespan as compared to the previous timespan. The "-" sign is used to indicate that less observables were contributed than during the previous timespan.
total	Total number of the contributions from your organization and community.

ioc_feeds

This macro allows you to query statistics of the observables imported from various feeds. When used in Search, it returns the fields containing information about feeds and observables, which then can be included in the operators for further analysis.

Fields	Description
total	Total number of observables contributed by the feed.
observables	Total number of observables contributed by the feed in the selected timespan.
feed_name	Name of the feed.
feed_categoty	This attribute is for internal use and can be ignored.
observables_ previous_period	Number of observables imported in the previous import. Not available for manual imports.
feed_id	The ID number for the feed
falsepositives	The Number of false positives in the observables in the selected timespan.

malwarethreats(limit)

This macro allows you to query statistics of the malware threats accessible via platform. When used in Search, it returns the fields containing information about malware, which then can be included in the operators for further analysis.

Fields	Description
created_ts	Time stamp of when the malware was created on ThreatStream, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
uuid	UUID (universally unique identifier) assigned to the malware for STIX compliance.
id	ID of the malware entity. This ID is automatically assigned to the entity by ThreatStream at the time of creation. You can use this ID to associate the malware entity with other Threat Model entities.

Fields	Description
modified_ts	Time stamp of when the malware was last updated on ThreatStream, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the malware. Malware name is associated with your organization. Therefore, the name must be unique within your organization. However, two malware with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
tags_v2	Additional information on the tags assigned to the Actor.
source_modified	Time stamp of when the malware was last modified by the source, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the malware. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing,exploitation.
tlp	Traffic Light Protocol designation for the malware—red, amber, amber+strict, green, clear.

Fields	Description
source_created	Time stamp of when the malware was originally created by the source, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
published_ts	Time stamp of when the malware was originally published by the source, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
publication_ status	Malware entities can be in new, pending_review, review_ requested, reviewed, or published statuses.
owner_user_id	Information about the user that created the entity on ThreatStream.
organization_id	Information about the organization that created the entity on ThreatStream.
last_seen	Time stamp of when the malware was last observed, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
killchainstages	For internal use only.
workgroups	Workgroups with which the malware is shared.

Fields	Description
is_public	Whether the malware is public or private (including belonging to a trusted circle). False—if the malware is private or belongs to a Trusted Circle True—if the malware is public Default: False
circles	All Trusted Circles to which the malware belongs.
intelligence_ initiatives	Intelligence Initiatives associated with the malware.
assignee_user	The user to whom the malware is assigned.
first_seen	Time stamp of when the malware was first observed, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
is_anonymous	Whether the malware user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
feed_id	Numeric ID of the threat feed that provided the malware entity.
resource_uri	Internal URL for the malware within ThreatStream.
aliases	Other names by which the malware is known.

rules(limit)

This macro allows you to query statistics of the rules configured in ThreatStream. When used in Search, it returns the fields containing information related to rules, which then can be included in the operators for further analysis.

Fields	Description
create_ investigation	Whether an investigation was created the first time the rule was triggered.
created_ts	Time stamp of when the rule was created, in UTC format.
description	Description for the rule.
exclude_notify_ org_whitelisted	Whether Observables whitelisted by your organization are excluded from rule matches.
exclude_notify_ owner_org	Whether keyword matches on Observables imported by your organization are excluded from Keyword Match or Hourly Digest email notifications.
has_associations	Whether the rule creates associations between specified entities and matched entities.
id	Unique ID assigned to the rule
intelligence_ initiatives	Intelligence Initiatives associated with the rule.
is_editable	If the current user have edit access to the rule
is_enabled	If the rule is enabled.
keywords	A list of keywords associated with the rule.
match_actors	Unused.
match_all_tm	Unused.
match_ attackpatterns	Unused.
match_ campaigns	Unused.
mathc_ courseofactions	Unused.
match_ customtms	Unused.

Fields	Description
match_identities	Unused.
match_incidents	Unused.
match_ infrastructures	Unused.
match_ intrusionsets	Unused.
match_malware	Unused.
match_ observables	Whether the rule matches keywords in newly created Observables.
match_ reportedfiles	Whether the rule matches keywords in newly created Sandbox Reports.
match_signatures	Whether the rule matches keywords in newly created Signatures.
match_tips	Whether the rule matches keywords in newly created Threat Bulletins.
match_ttps	Unused.
match_ vulnerabilities	Whether the rule matches keywords in newly created Vulnerabilities.
matches	Total number of keyword matches for the rule.
modified_ts	Time stamp of when the rule was last modified, in UTC format.
name	Name of the rule.
org_id	ID associated with the organization that created the rule.
org_shared	Unused.
organization	ID, Name, and Resource URI associated with the organization that created the rule.

Fields	Description
resource_uri	Resource URI associated with the rule.
tags	Tags applied to matched entities.
user	User that created the rule.
user_id	ID of the user that created the rule.
workgroups	Workgroups associated with the rule.

signature(limit)

This macro allows you to query the data related to Signatures. When used in Search, it returns the fields containing Signature details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the Signature is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Signature.
circles	All trusted circles to which the Signature belongs.
created_ts	Time stamp of when the Signature was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
end_date	The time when this Signature is known to have ended. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Signature.
id	ID of the Signature. This ID is automatically assigned to the entity by ThreatStream at the time of creation.

Fields	Description
intelligence_initiatives	Intelligence Initiatives associated with the Signature.
is_anonymous	Indicates whether the Signature user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Signature can be cloned.
is_public	Indicates whether the Signature is public or private (including belonging to a trusted circle). False—if the Signature is private or belongs to a trusted circle True—if the Signature is public
modified_ts	Timestamp of when the Signature was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Signature. The Signature name is associated with your organization. Therefore, the name must be unique within your organization. However, two Signatures with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Signature on ThreatStream.
owner_user_id	ID of the user that created the Signature on ThreatStream.
publication_status	Signature entities can be in new, pending_review, review_requested, reviewed, or published statuses.

Fields	Description
published_ts	Timestamp of when the Signature was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Signature entity on ThreatStream.
s_type	Signature type.
source_created	Timestamp to when the Signature was first created.
source_modified	Timestamp to when the Signature was last updated.
tags	Tags assigned to the Signature. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Signature.
target_industry	Target industries associated with the Signature.
tlp	Traffic Light Protocol assigned to the Signature. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Signature for STIX compliance.
workgroups	Workgroups to which the Signature is visible.

tipreport(limit)

This macro allows you to query the data related to Threat Bulletins. When used in Search, it returns the fields containing Threat Bulletin details, which then can be included in the operators for further analysis.

Fields	Description
all_circles_visible	Indicates whether the Threat Bulletin is visible to all trusted circles.
assignee_org	Organization to which the Threat Bulletin is assigned
assignee_org_id	ID of the organization to which the Threat Bulletin is assigned.
assignee_org_name	Name of the organization to which the Threat Bulletin is assigned.
assignee_user	User to which the Threat Bulletin is assigned.
assignee_user_id	ID of the user to which the Threat Bulletin is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Threat Bulletin.
circles	All trusted circles to which the Threat Bulletin belongs.
created_ts	Time stamp of when the Threat Bulletin was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Threat Bulletin.
id	ID of the Threat Bulletin. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Threat Bulletin.
is_anonymous	Indicates whether the Threat Bulletin user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Threat Bulletin can be cloned.

Fields	Description
is_email	Indicates whether the Threat Bulletin is email.
is_public	Indicates whether the Threat Bulletin is public or private (including belonging to a trusted circle). False—if the Threat Bulletin is private or belongs to a trusted circle True—if the Threat Bulletin is public
locations	Locations associated with the Threat Bulletin.
modified_ts	Timestamp of when the Threat Bulletin was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Threat Bulletin. The Threat Bulletin name is associated with your organization. Therefore, the name must be unique within your organization. However, two Threat Bulletins with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
original_source	Original source from where the Threat Bulletin came.
original_source_id	ID of the original source of the Threat Bulletin.
owner_org	Organization owner of the Threat Bulletin.
owner_org_id	ID of the organization owning the Threat Bulletin.
owner_org_name	Name of the organization owning the Threat Bulletin.
parent	For internal use only.
publication_status	Signature entities can be in new, pending_review, review_requested, reviewed, or published statuses.

Fields	Description
published_ts	Timestamp of when the Threat Bulletin was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Threat Bulletin on ThreatStream.
source	Source of the Threat Bulletin intelligence.
source_created	Timestamp to when the Threat Bulletin was first created.
source_locations	Source locationa associated with the Threat Bulletin (as defined by STIX 2.1).
source_modified	Timestamp to when the Threat Bulletin was last updated.
status	Provides the status of the Threat Bulletin. Possible values—published, reviewed, review_ requested, pending_review.
tags	Tags assigned to the Threat Bulletin. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Threat Bulletin.
target_industry	Target industries associated with the Threat Bulletin.
threat_actor	Actors associated with the Threat Bulletin.
tlp	Traffic Light Protocol assigned to the Threat Bulletin. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Threat Bulletin for STIX compliance.
workgroups	Workgroups to which the Threat Bulletin is visible.

ttp(limit)

This macro allows you to query the data related to TTPs. When used in Search, it returns the fields containing TTPs details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the TTP is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the TTP.
circles	All trusted circles to which the TTP belongs.
created_ts	Time stamp of when the TTP was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the TTP.
id	ID of the TTP. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the TTP.
is_anonymous	Indicates whether the TTP user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_category	Indicates whether a category is assigned to TTP.
is_cloneable	Indicates whether the TTP can be cloned.
is_mitre	Indicates whether the TTP is described in the MITRE list. True—if the TTP is described False—if the TTP is not described

Fields	Description
is_public	Indicates whether the TTP is public or private (including belonging to a trusted circle). False—if the TTP is private or belongs to a trusted circle True—if the TTP is public
modified_ts	Timestamp of when the TTP was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the TTP. The TTP name is associated with your organization. Therefore, the name must be unique within your organization. However, two TTPs with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the TTP on ThreatStream.
publication_status	TTP entities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Timestamp of when the TTP was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the TTP entity on ThreatStream.
source_created	Timestamp to when the TTP was first created.
source_modified	Timestamp to when the TTP was last updated.

Fields	Description
tags	Tags assigned to the TTP. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the TTP.
target_industry	Target industries associated with the TTP.
tlp	Traffic Light Protocol assigned to the TTP. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the TTP for STIX compliance.
workgroups	Workgroups to which the TTP is visible.

vulnerability(limit)

This macro allows you to query the data related to Vulnerabilities. When used in Search, it returns the fields containing Vulnerability details, which then can be included in the operators for further analysis.

Fields	Description
aliases	Other names associated with the Vulnerability.
assignee_user	ID of the user to which the Vulnerability is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Vulnerability.
circles	All trusted circles to which the Vulnerability belongs.
created_ts	Time stamp of when the Vulnerability was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.

Fields	Description
feed_id	Numeric ID of the threat feed that provided the Vulnerability.
id	ID of the Vulnerability. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Vulnerability.
is_anonymous	Indicates whether the Vulnerability user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Vulnerability can be cloned.
is_public	Indicates whether the Vulnerability is public or private (including belonging to a trusted circle). False—if the Vulnerability is private or belongs to a trusted circle True—if the Vulnerability is public
is_system	Indicates whether the Vulnerability was imported by ThreatStream. True—if the Vulnerability was imported by ThreatStream False—if the Vulnerability was added by a ThreatStream user
modified_ts	Timestamp of when the Vulnerability was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.

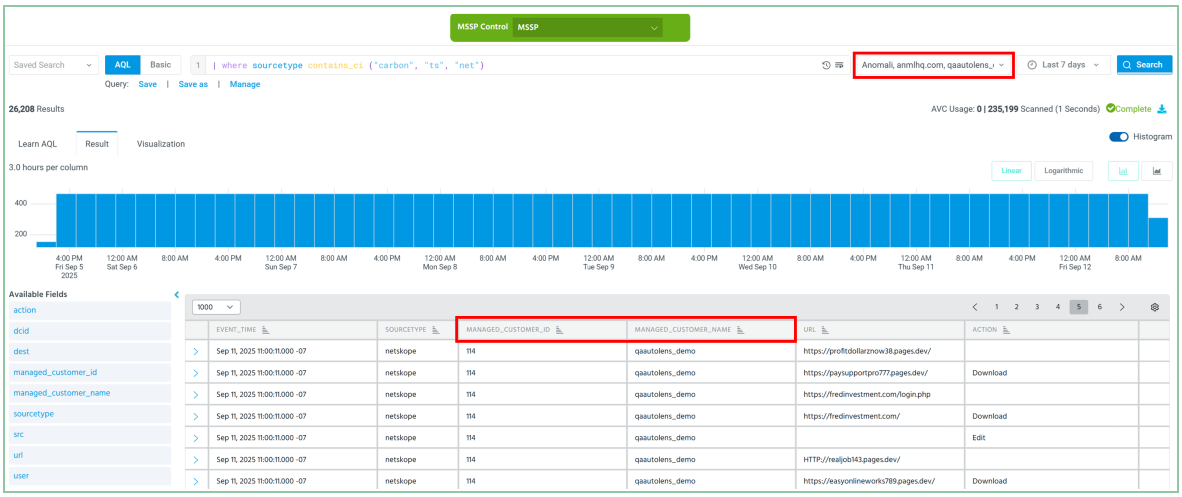
Fields	Description
name	Name of the Vulnerability. The Vulnerability name is associated with your organization. Therefore, the name must be unique within your organization. However, two Vulnerabilities with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Vulnerability on ThreatStream.
owner_user_id	ID of the user that created the Vulnerability on ThreatStream.
publication_status	Vulnerabilities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Timestamp of when the Vulnerability was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Vulnerability entity on ThreatStream.
source	Source from where the Vulnerability was imported from.
source_created	Timestamp to when the Vulnerability was first created.
source_modified	Timestamp to when the Vulnerability was last updated.
tags	Tags assigned to the Vulnerability. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Vulnerability.

Fields	Description
target_industry	Target industries associated with the Vulnerability.
tlp	Traffic Light Protocol assigned to the Vulnerability. Possible values: red, amber, amber+strict, green, and clear.
update_id	Updated Vulnerability ID.
uuid	UUID (universally unique identifier) assigned to the Vulnerability for STIX compliance.
workgroups	Workgroups to which the Vulnerability is visible.

Chapter 13: Federated Search

Federated search enables you to query and retrieve data across multiple managed organizations (tenants) from a single interface. The Anomali platform supports federated search by leveraging the Managed Security Service Providers (MSSP) feature, which allows your organization to manage its customers as multiple sub-organizations, streamlining data, features, investigations, and reporting while maintaining data isolation between customers.

How federated search works



As an MSSP org admin, federated search allows you to search across multiple customers at once, such that each sub-organization's data remains separate and is not shared across other organizations.

To ensure unique identification, a federated search adds two federated fields to the search results:

- `managed_customer_id`: A string that denotes the organization ID of the managed customer where the data originated.
- `managed_customer_name`: A string that displays the name of the managed customer.

These federated fields can be referenced in queries to differentiate results by organization.

Additionally, an an MSSP org admin, you can also use the dropdown next to the Search bar to filter the results specific to a sub-organization. Selecting such a sub-organization from the list assumes the environment, users, results, and settings specific only to that sub-organization.

Best Practices

Lookup Tables

You can use the federated fields in your AQL query like any regular query, since these fields are included in the eventlog schema as well as all lookup table.

Subsearch

When running a federated search with multiple managed organizations, you cannot use the RIGHT join. Anomali recommends that you either use a LEFT join, an INNER join, or reduce the search scope to only one organization.

Examples

Example 1: As an MSSP, issue a federated search like any regular AQL Search query to see consolidated results across multiple sub-orgs.

```
| where sourcetype contains_ci ("ts", "carbon", "nets")
```

Example 2: federated fields in an AQL query

```
|aggr count by managed_customer_name
```

Chapter 14: Forensic and Retrospective Search

Searching through months or years of event data to find an indicator match can be a time consuming and resource-intensive process. Anomali Search provides the `turbosearch` operator that is purpose-built to rapidly search through large datasets and identify malicious indicators in your events that were logged in the past. This operator is ideal for examining event data across large time spans such as the last 7 days, 30 days, or up to 90 days.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

> **Note:** Similar to all search operations on Security Analytics, the `turbosearch` operator searches events that fall within the data retention policy associated with your subscription license. See Anomali Search Data Retention Policy to understand how your data is retained over time in the Anomali platform.

Use cases

You can use the `turbosearch` operator for the following applications:

- Forensic or retrospective searches
- Threat hunting
- Needle in a haystack scenarios

See the turbosearch operator documentation for more information on its syntax and the different threat indicator types it supports.

Chapter 15: About AQL Visualization Types

Anomali Query Language (AQL) dashboards support multiple visualization types. These visualizations support both the eventlog and OCSF Version 1.2 schema fields with dotted notation. For more information on using OCSF fields in queries and visualizations, see "OCSF Schema Overview" on page 39.

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Types of Visualizations

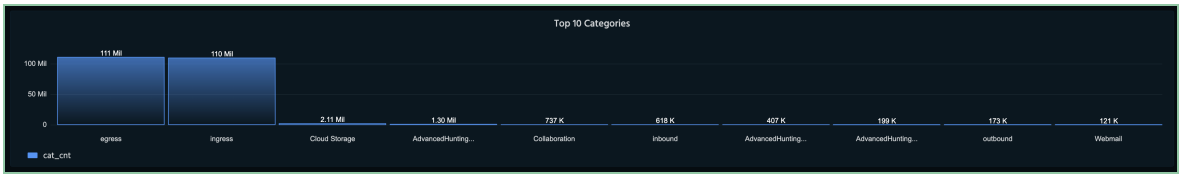
About the Bar Chart Visualization Type 253
About Base64 Visualization Type 266
About the Bubble Chart Visualization Type 270
About the Copilot Widget 273
- Key features 274
- Use cases 274
- Using Copilot in Dashboards 276

About the Decision Tree Visualization Type 277
About the Heatmap Visualization Type 280
About the Gauge Visualization Type 288
About the Geo Map Visualization Type 298
About the Gantt Time Series Visualization Type 301
About the Linear Regression Visualization Type 310

About the MITRE ATT&CK Flow Visualization Type 313
Panel Options 314
Exporting Attack Flows 316
About the MITRE ATT&CK Heatmap Visualization Type 316
Panel Options 318
About the Pie Chart Visualization Type 327
About the Sankey Diagram Visualization Type 337
About the Stat Visualization Type 339
About the Table Visualization Type 349
About the Text Visualization Type 363
Support for Variables 370
About the Timeline Visualization Type 373
Support for Variables 381
About the TimeSeries Visualization Type 384
About the Wordcloud Visualization Type 394

About the Bar Chart Visualization Type

Use the Bar Chart visualization type to show aggregate counts of values for a single field.



AQL Expression Requirements

The bar chart visualization type is used to show counts. The query must have the following elements to return the required data:

- A result set having a string column and a numeric column.
- Use the `aggr` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

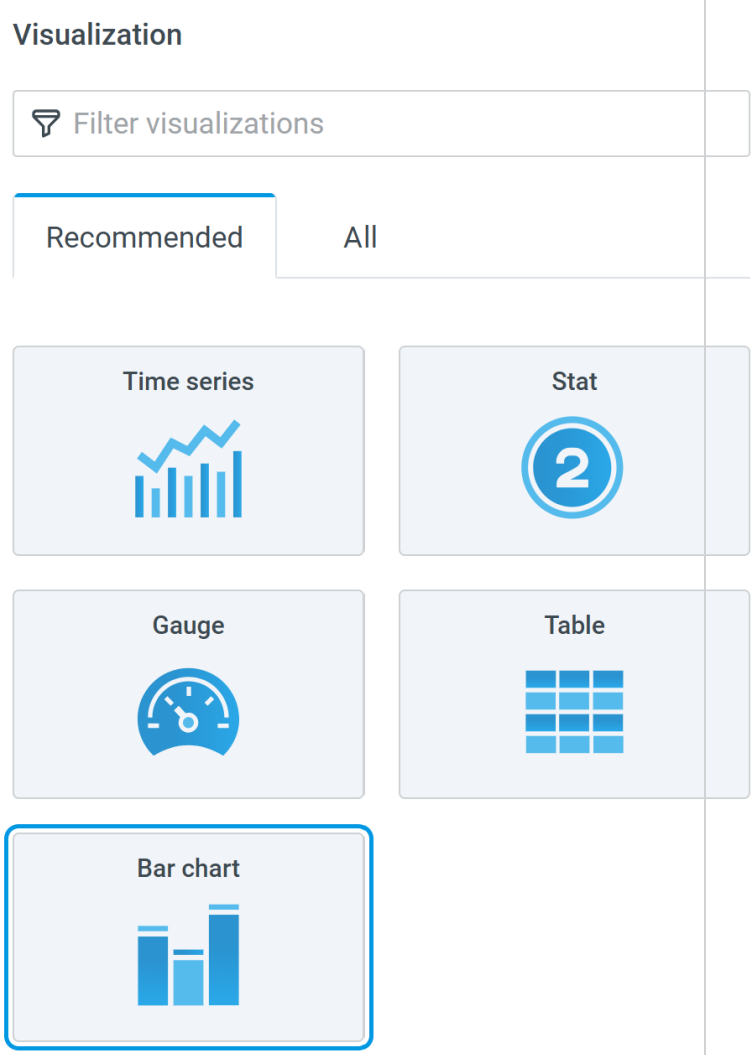
```
| aggr count as cnt by sourcetype | sort cnt desc | limit 10
```

Note: Using `addtotals=true` in your timechart operator query appends a total row and total column to your results. The `total` row shows the aggregate count per group across the entire time range, and the `total` column shows the aggregate count across all groups per time bucket. This is useful for comparing entities and spotting volume trends, without needing a separate query. See timechart operator for more information.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
	Show Description	Display or hide the description.

Setting		Description
Visualization	Bar chart	Select the Bar chart widget to create bar charts. 

Setting		Description
Display	Orientation	• Auto • Horizontal: Count is x-axis. • Vertical: Count is y-axis.
	Rotate x-axis tick labels	Use the slider to rotate the label text as needed to enhance readability.
	Stacking	Select whether to stack bars in the chart.
	Show values	Show counts above or at the end of bars. • Auto • Always • Min/Max • Never
	Value format	Select whether to display the full bar value or its nearest approximate value.
	Group width	Use the slider to adjust the width of bar groups.
	Bar width	Use the slider to adjust the width of bars.
	Show decimals	Show decimals in the values.
	X Axis	Select the field to be mapped on the x-axis. By default, no field is selected. When no field is selected, the widget automatically displays all the available series data for all the fields.
	Hide series	Select one or more fields that you would like to hide.

Setting		Description
	Legend mode	• List • Table • Hidden
	Legend placement	• Bottom • Right
	Legend calculations	Select functions. Advanced users only.

Field Options

Use the options in the Fields tab to customize the options described in the following table.

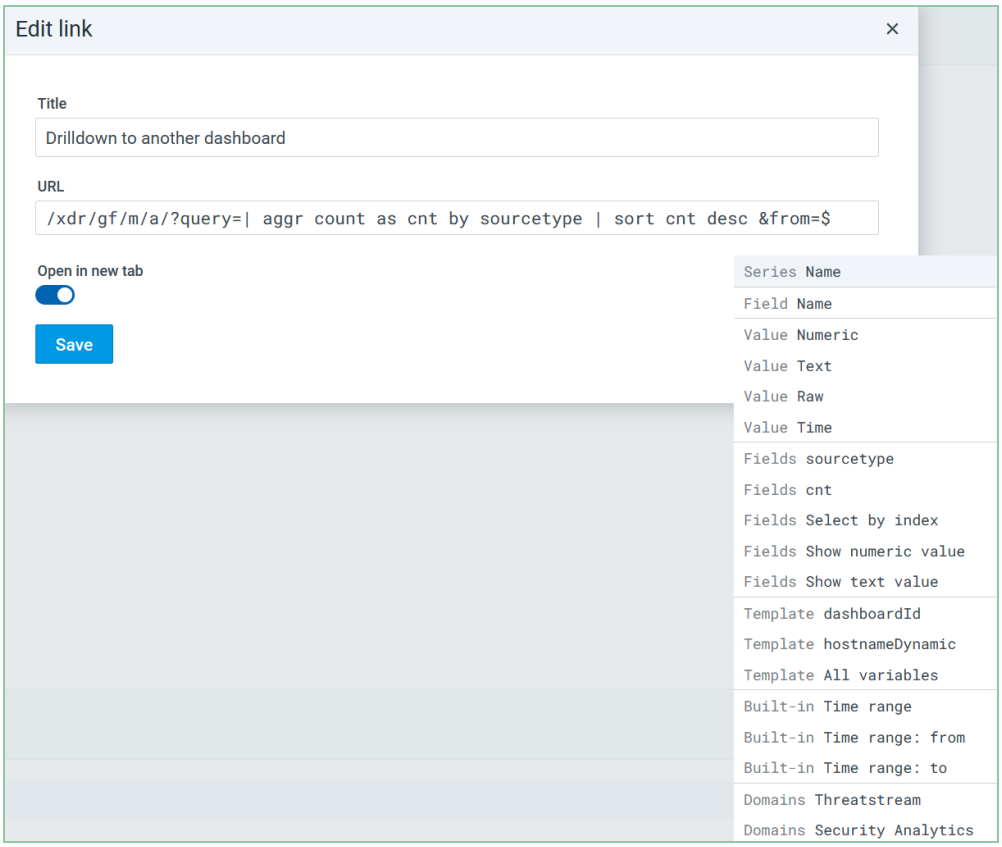
Setting		Description
Bar chart options	Line width	Use the slider to adjust the bar border width.
	Fill opacity	Use the slide to adjust the bar fill.
	Gradient mode	• None • Opacity • Hue Sensitivity of this setting is affected by fill opacity and line color.

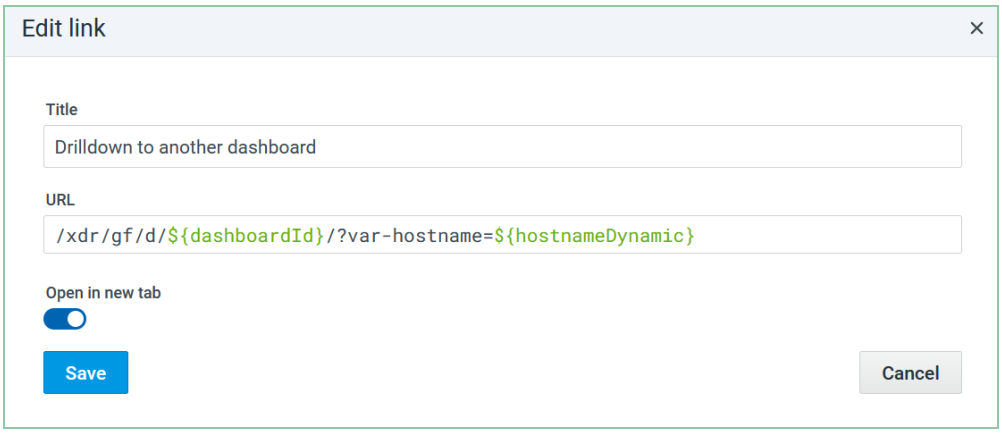
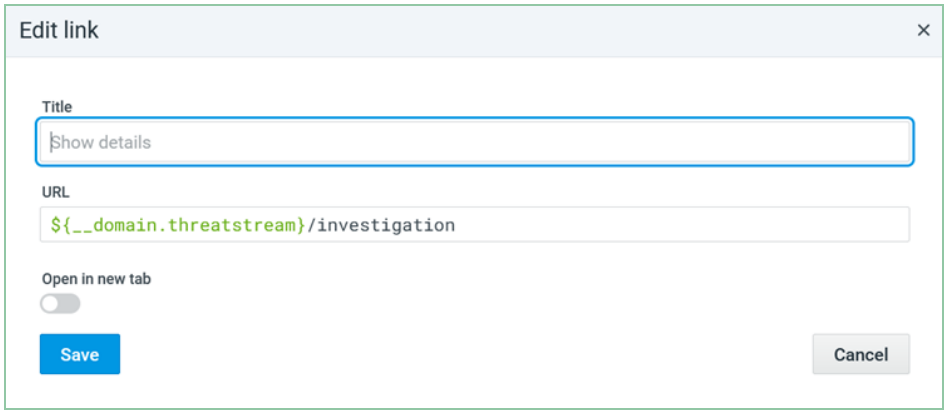
Setting		Description
Axis	Placement	• Auto • Left • Right • Hidden
	Label	Additional label for y-axis.
	Width	By default, the width is calculated automatically based on the results data. If desired, specify a fixed width instead.
	Softmin Softmax	By default, the Y-axis limits are determined automatically based on the results data. Adjust these settings if the automatic behavior distorts the visualization.
Thresholds	Add threshold	Configure mapping of thresholds to colors.
	Thresholds mode	• Absolute • Percentage

Setting		Description
Standard options	Unit	Select a unit. Applies to all fields except time.
	Min	Specify a minimum value (as percentile). Leave empty to calculate based on all values.
	Max	Specify a maximum value (as percentile). Leave empty to calculate based on all values.
	Decimals	Number of decimal places in numeric values.
	Displayname	Change the field or series name.
	Novalue	What to show when there is no value.
	Colorscheme	Specify a single color or use the color palette.

Setting		Description
Value mappings	Add value mapping	Click **+Add value mapping** to map a single value or range of values to a specified text string. For example, map data having the country code `SG` to the text `Singapore`.

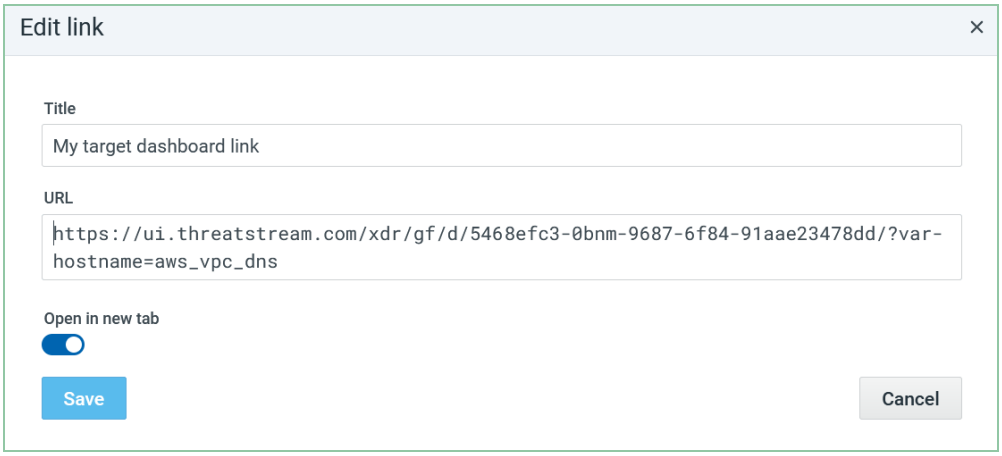
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

About Base64 Visualization Type

Use the Base64 visualization type to add images as well as audio, video, and PDF files to your dashboards.

Supported Formats

With the Base64 widget, you can add the following media formats:

- Images: JPEG, GIF, PNG, HEIC
- PDF files
- Video: MP4, WEBM
- Audio: MP3, OGG

Panel Options

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

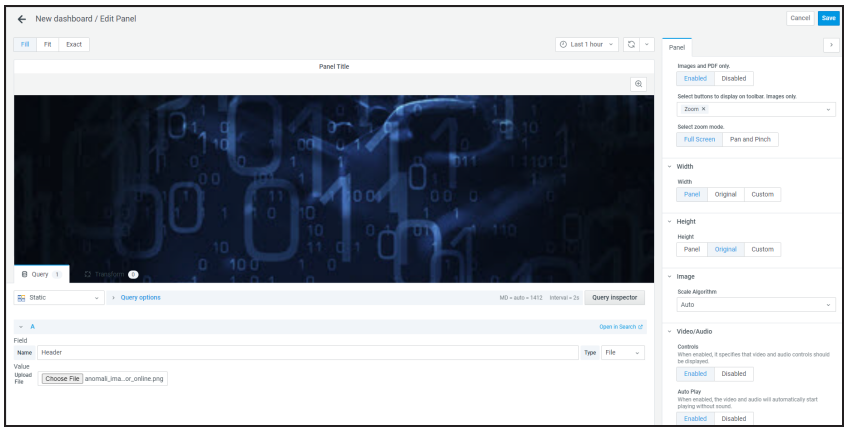
Setting		Description
Display	No Results Message	Provide the message that will appear if media files are not displayed.
Toolbar	Images and PDF only	Specify whether to display a toolbar for images and PDFs: • Enabled • Disabled
	Select buttons to display on toolbar. Images only.	If the Images and PDF only option is enabled, add buttons for working with images: • Navigation • Download • Zoom
	Select zoom mode	If you added the Zoom button for images, select the zoom mode: • Full Screen • Pin and Pinch
Width	Width	Select the file width: • Panel • Original • Custom
	Custom width (px)	If you selected the custom width, specify its value in pixels.

Setting		Description
Height	Height	Select the file height: • Panel • Original • Custom
	Custom width (px)	If you selected the custom height, specify its value in pixels.
Image	Scale Algorithm	Select the scaling algorithm: • Auto • Crisp Edges • Pixellated
Video/Audio	Controls	Specify whether audio and video controls should be displayed: • Enabled • Disabled
	Autoplay	Specify whether audio and video should start automatically playing without the sound: • Enabled • Disabled
	Infinity play	Specify whether audio and video should be played on the loop: • Enabled • Disabled

Configuring a Dashboard Panel for Base64 Visualization Type

1. Add an empty panel to your dashboard. For more information, see .
2. Click **Base64** in the Panel tab.
3. Select **Static** in the Query tab, and then do the following:
   a. Specify the name of the media file that will be displayed on the dashboard.
   b. Select the file type from the **Type** list, and depending on the selected file type, specify the value:
      - String: Provide a Base64-encoded string representing the file.
      - File: Upload the media file from your PC.
      - Url: Specify the URL of the media file.

> **Note:** When uploading the file, make sure that the file is safe and its content is relevant to the dashboard. During and after the upload, Anomali reserves the rights to remove the files that fail to meet security, relevancy, and other standards.



The file is displayed on the panel.

4. Customize the panel in the Panel tab as needed. For a complete list of panel

settings, see "Panel Options" on page 266.

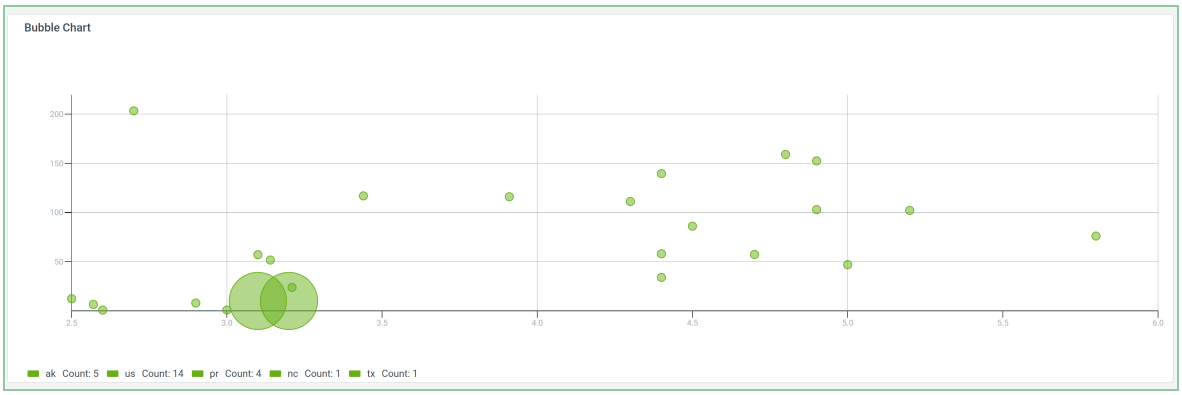
5. Save your changes.

A panel containing the media file is added to the dashboard.

About the Bubble Chart Visualization Type

Use the Bubble Chart visualization type to represent three dimensions of data simultaneously, within a single chart. This capability is particularly valuable for identifying patterns, outliers, and relationships across complex datasets.

For example, bubble charts can be used to correlate multiple metrics, such as event frequency, severity, and affected assets, helping you gain actionable insights at a glance. In these charts, the bubble position represents two dimensions of the series data, while the bubble size represents the third dimension.



Use case

You can use bubble charts to:

- Analyze security events by type, frequency, and severity. In this case, the X-axis can be the event type (phishing, malware, and so on), Y-axis can be the number of occurrences, and the bubble size indicates the event severity or impact score of each event type. Additionally, color coding could indicate the resolution status: resolved or unresolved.

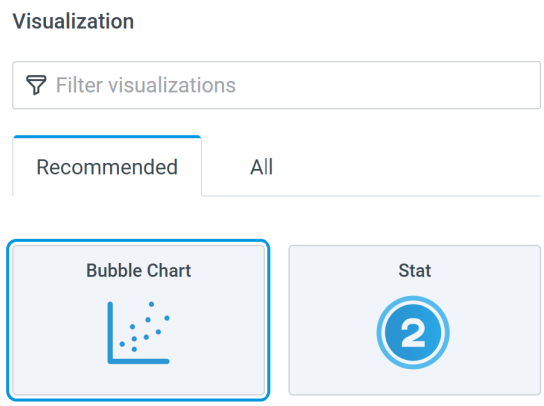
- Assess which assets (servers, endpoints, applications) are most at risk based on recent security findings. Here, the X-axis can represent asset categories or specific assets, Y-axis can represent the number of vulnerabilities or incidents detected, and the bubble size could reflect the criticality of the asset (for example,

business importance or data sensitivity).

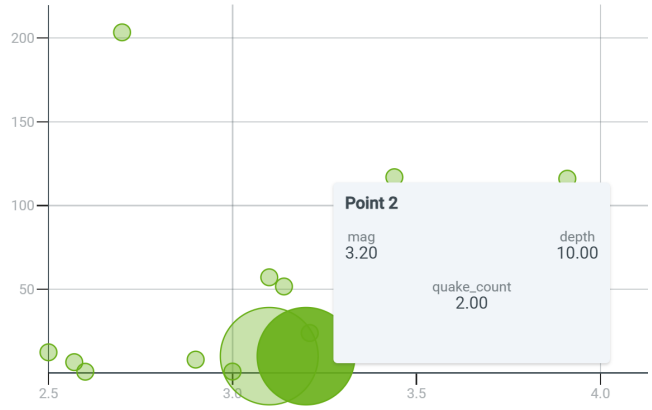
- Detect anomalous user behavior that may indicate insider threats or compromised accounts. In this, the X-axis can represent user accounts or departments, the Y-axis can represent the number of suspicious activities or policy violations, and the bubble size can indicate the cumulative risk score or the number of unique incident types per user.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
	Show Description	Display or hide the description.
Visualization	Bar chart	Select the Bar chart widget to create bar charts. 

Setting		Description
Display	X-axis Field	Select the field to display as the horizontal axis of the chart.
	Y-axis Field	Select the field to display as the vertical axis of the chart.
	Bubble Size Field	Select the field that determines the size of the bubble. Typically, this represents a relative quantitative measure of a certain entity, such as event severity, impact score, or number of vulnerabilities.
	Color Field	Select the field by which the chart bubbles are colored. Typically, this color represents the relative magnitude of an entity, such as the resolved or unresolved status of an alert.
	Min Bubble Size	The relative size of the smallest bubble in the chart.
	Max Bubble Size	The relative size of the largest bubble in the chart.
	Fill Opacity	Select the transparency of the bubbles, with 0 for transparent bubbles and 100 for opaque colored bubbles.
	Stroke Width	Select the thickness of the perimeter of the bubble.
	X Axis Label	Select the label of the X-axis.
	Y Axis Label	Select the label of the Y-axis.

Setting		Description
	Show Tooltip	Toggle to show or hide the chart tooltip when you hover your mouse. 
	Legend mode	Select whether to display the legend as a list, table, or to hide it completely.
	Legend placement	Select whether to place the legend at the bottom or right-hand side of the chart.
	Legend calculations	Select a reducer function such as minimum value, maximum value, mean, number of distinct values, or first non-null value, and more.

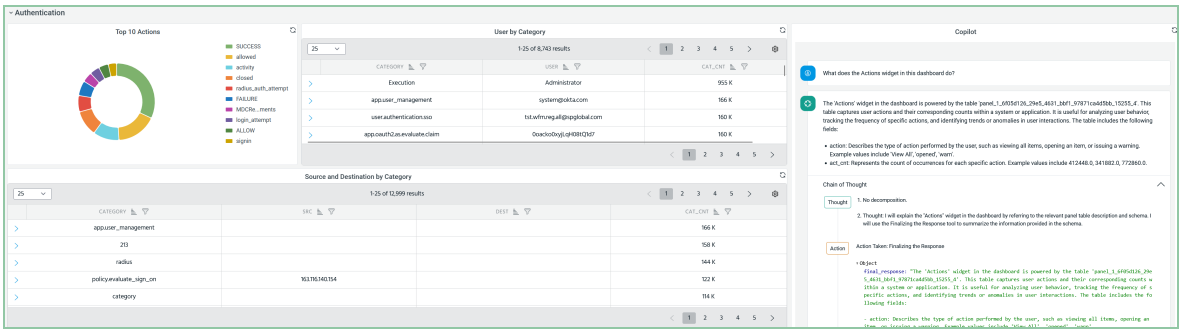
About the Copilot Widget

Use the Copilot widget to gather AI-powered insights on threat hunting, enrichments, integrators, and more, based on data in a dashboard. With the ability to summarize different panels in a dashboard and communicate its chain of thought in a user-friendly chat interface, the Copilot widget represents a comprehensive set of features that security analysts can use to analyze different types of dashboard data, associated trends, and perform advanced analysis.

The Copilot widget leverages the underlying intelligence capabilities of an AI model that employs a Chain-of-Thought process that ensures user transparency by allowing you to examine the sources and thought patterns the AI agent uses to answer your question.

For example, you can analyze the results of a widget by asking Copilot questions about what the widget does, what its results signify, or perform further analysis

through more questions about the dashboard data. In the following example, the user asks Copilot about what the Actions widget does and what its results signify.



Key features

As a security analyst, you can chat with Copilot within a dashboard to get insights without switching contexts. The Chain of Thought process provides clear transparency on all the context from the dashboard the AI model uses to construct its response.

> **Note:** Copilot preserves the context of a conversation within a session. The widget will reference your current chat transcript in answering your question. The data is not preserved between sessions.

Use cases

The Copilot widget supports a broad range of use cases:

Generate Reports on IOCs:

Copilot leverages its knowledge on any ThreatStream observables, including URLs, file hashes, domains, IP addresses, or emails. Based on this knowledge, it generates summary reports about these observables and their relevant association within the data in the panels of the dashboard that contains this Copilot widget.

This association, therefore, generates sophisticated summaries that provide you with a comprehensive snapshot of ThreatStream observables and related information across all your dashboard panels for further analysis.

Data Enrichment using ThreatStream

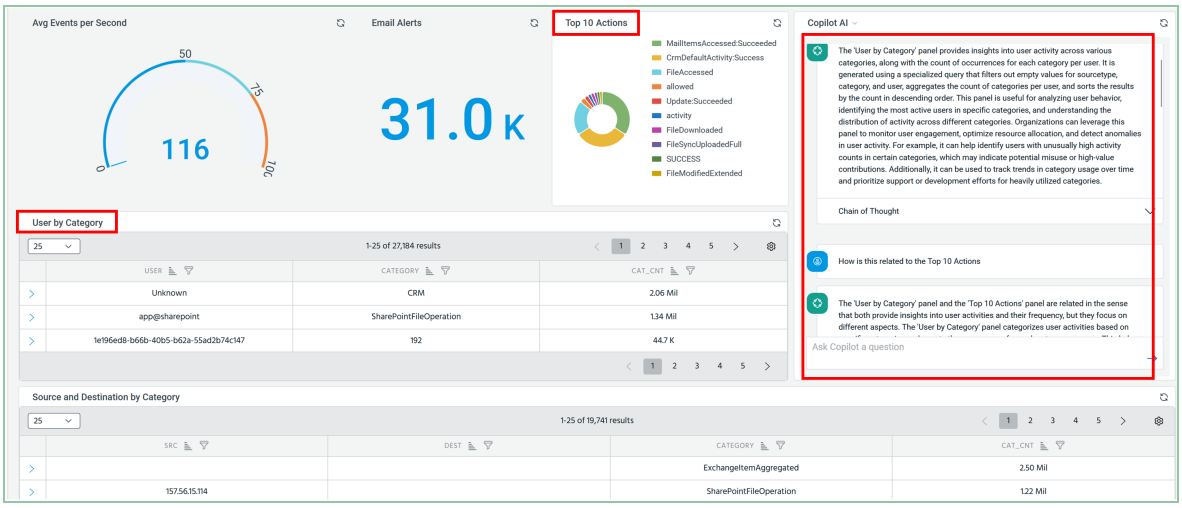
Copilot uses ThreatStream to answer any question about threat models and intelligence related to your dashboard data, and deliver precise responses by leveraging Anomali's resources backed by verified sources and searches.

The dashboard panel agent can ingest, manage, and analyze contextual, relationship, and workflow information for various threat model entities such as actors, malware, infrastructure, observables, and vulnerabilities, so as to:

- Provide more information about threat actors and their TTPs.
- Perform enrichment by ingesting threat intelligence data from Anomali ThreatStream, as well as external sources such as VirusTotal.
- Identify threats and enable proactive threat management.
- Support SOC analysts in monitoring and defending against threats by centralizing and using aggregated and contextualized threat intelligence.

Therefore, Copilot enables you to quickly access contextual information about your dashboard data and all its different associations across ThreatStream, while eliminating the need for manual observable or threat model entity searches.

In the following example, the user asks Copilot how the Top 10 Actions are associated with the User by Category, to which Copilot responds that while the Top 10 Actions panel captures user behavior trends, the latter focuses on the actual network and system interactions.

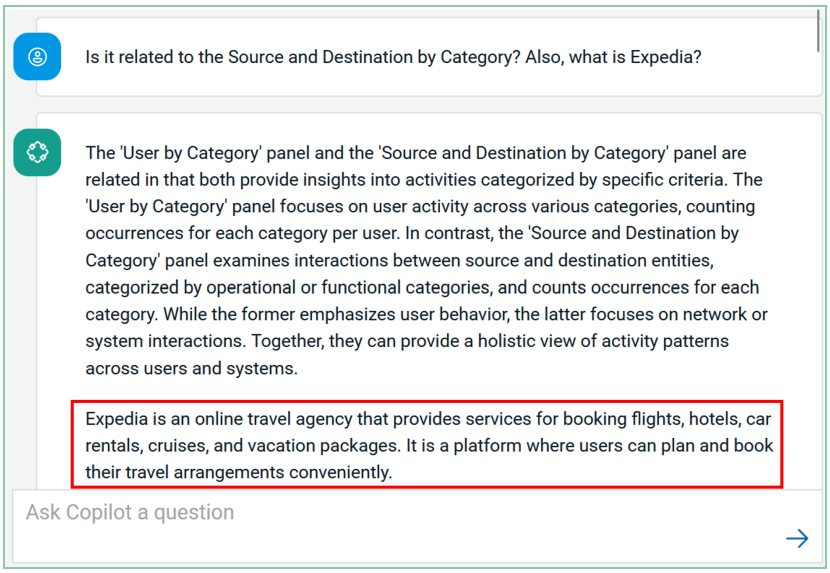


Retrieve Trending Threat Entities

Copilot also allows you to retrieve trending threat entities that provide greater insight into the latest threats and help you strategize incident management. With the ability to capture potential malicious actors, attack patterns, CVEs, or malware, Copilot helps you improve the overall threat management of your organization.

Internet search

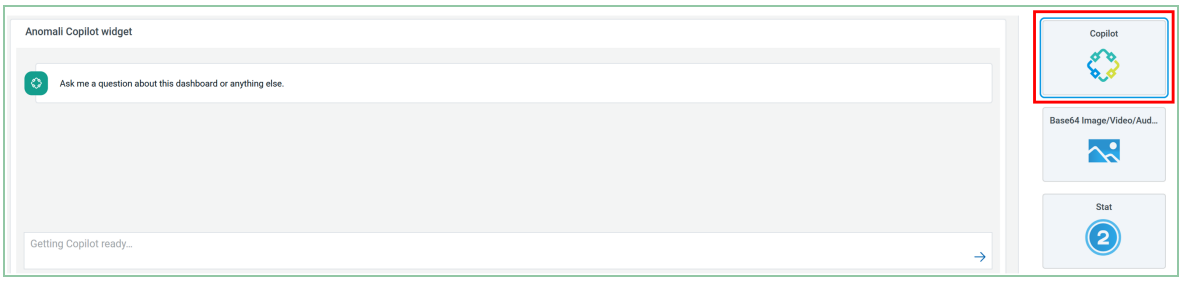
Copilot also leverages the world wide web to answer any question that is possibly irrelevant to cybersecurity or those that it cannot answer using ThreatStream alone.



Using Copilot in Dashboards

> **Note:** A dashboard can contain only one Copilot widget.

Within your dashboard, add the Copilot widget as a panel in the dashboard.



Use the options in the Panel tab to customize the options described in the following table.

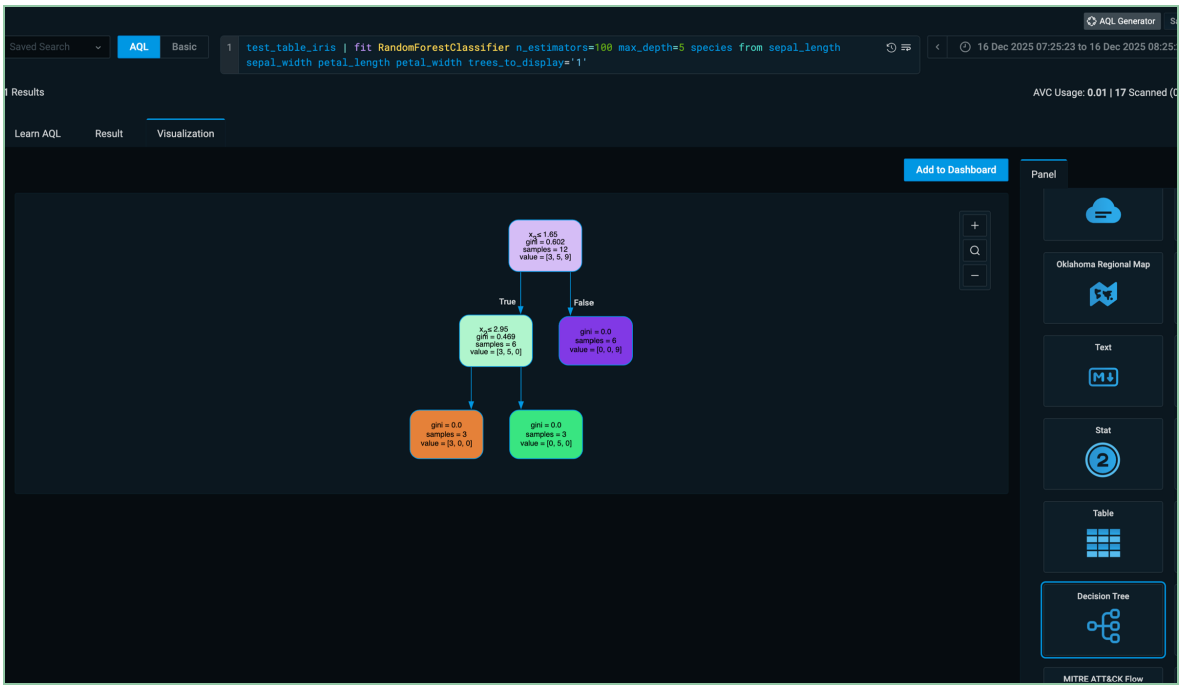
Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
	Show Description	Toggle whether to show the description below the panel title.

About the Decision Tree Visualization Type

Use the Decision Tree visualization type to intuitively explore and interpret tree-based machine learning models directly within the platform. By providing interactive, graphical representations of decision trees, this feature empowers analysts and data scientists to better understand model logic, validate outcomes, and communicate insights to stakeholders—all without leaving the platform.

This visualization type enables you to examine decision paths, feature importance, and node details, supporting both transparency as well as explainability in machine learning operations.

> **Note:** The Decision Tree visualization type supports both the `DecisionTreeRegressor` and the `RandomForestClassifier` as AQL keywords.



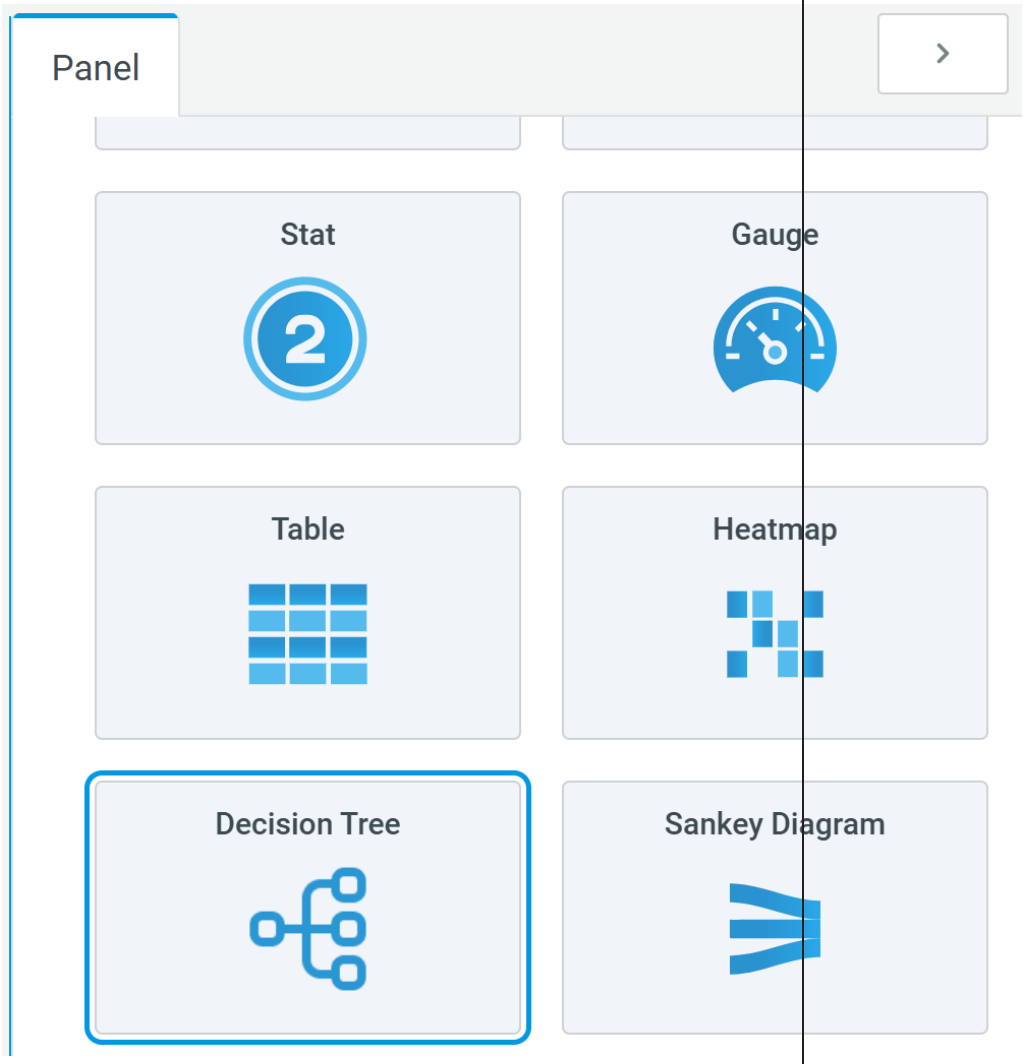
Use case

You can use decision trees to:

- **Model Interpretation and Validation**: Quickly understand how a decision tree model makes predictions by visually tracing decision paths and examining node splits. This transparency aids in debugging, model validation, and compliance with explainable AI requirements.
- **Feature Importance Analysis**: Identify which features most influence model outcomes by inspecting split criteria and feature importance metrics, supporting feature engineering and model refinement.
- **Regulatory and Audit Support**: Provide visual evidence of model logic and decision criteria to satisfy regulatory requirements for regulatory transparency in AI systems.
- **Stakeholder Communication**: Generate clear, shareable visualizations that help non-technical stakeholders grasp the logic behind model decisions, fostering user reliability and facilitating informed decision-making.

Panel Options

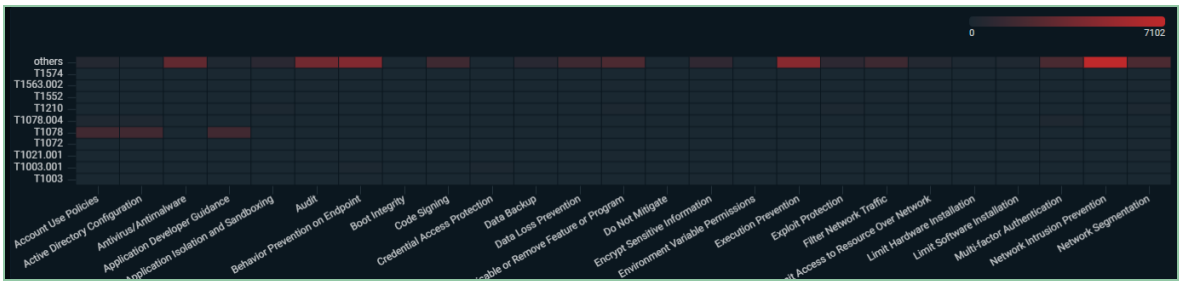
Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
Visualization	Decision Tree	Select the Decision Tree visualization to create decision trees. 

Setting		Description
Display	Show Feature Names	Toggles display of feature names in decision nodes.
	Show Class Labels	Toggles display of class labels in the leaf nodes.
	Show Impurity Measures	Toggles display of impurity measures (gini, entropy, squared_error).
	Show Sample Counts	Toggles display of the number of samples in each node.
	Max Depth	Sets the maximum depth of the tree to display.
	Node Shape	Defines the shape of the tree nodes.
	Font Size	Sets the font size for node labels.
	Tree Direction	Specifies the layout direction of the tree.
	Export File Name	Sets the default file name for exports.
	Enable Zoom	Toggles allowing zooming in or out of the tree.
	Show Tooltips	Displays tooltips on hovering over the node.

About the Heatmap Visualization Type

Use the Heatmap visualization type to quickly identify patterns, trends, and anomalies by displaying the data in a form of a matrix, where values are represented by varying colors.



You can use heatmaps to:

- Plot the volume of traffic between different IP addresses over time. This can help you spot potential DDoS attacks or unusual data exfiltration.
- Show the number of failed login attempts across different user accounts and time periods and, as a result, highlight hotspots of suspicious activity.
- Plot the volume and types of user activities across different departments or roles in order to identify unusual behavior and therefore, detect insider threats or policy violations.

AQL Expression Requirements

The heatmap visualization type is used to show counts. The query must have the following elements to return the required data:

- A result set in a form of a matrix with numeric values.
- Use the `aggr` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

```
techniques
| where name in ($name)
| fields system_requirements as mitigation, technique
| chart count as cnt by mitigation,technique
```

> **Note:** Using `addtotals=true` in your timechart operator query appends a total row and total column to your results. The `total` row shows the aggregate count per group across the entire time

> range, and the `total` column shows the aggregate count across all groups per time bucket. This is useful for comparing entities and spotting volume trends, without needing a separate query. See timechart operator for more information.

Panel Options

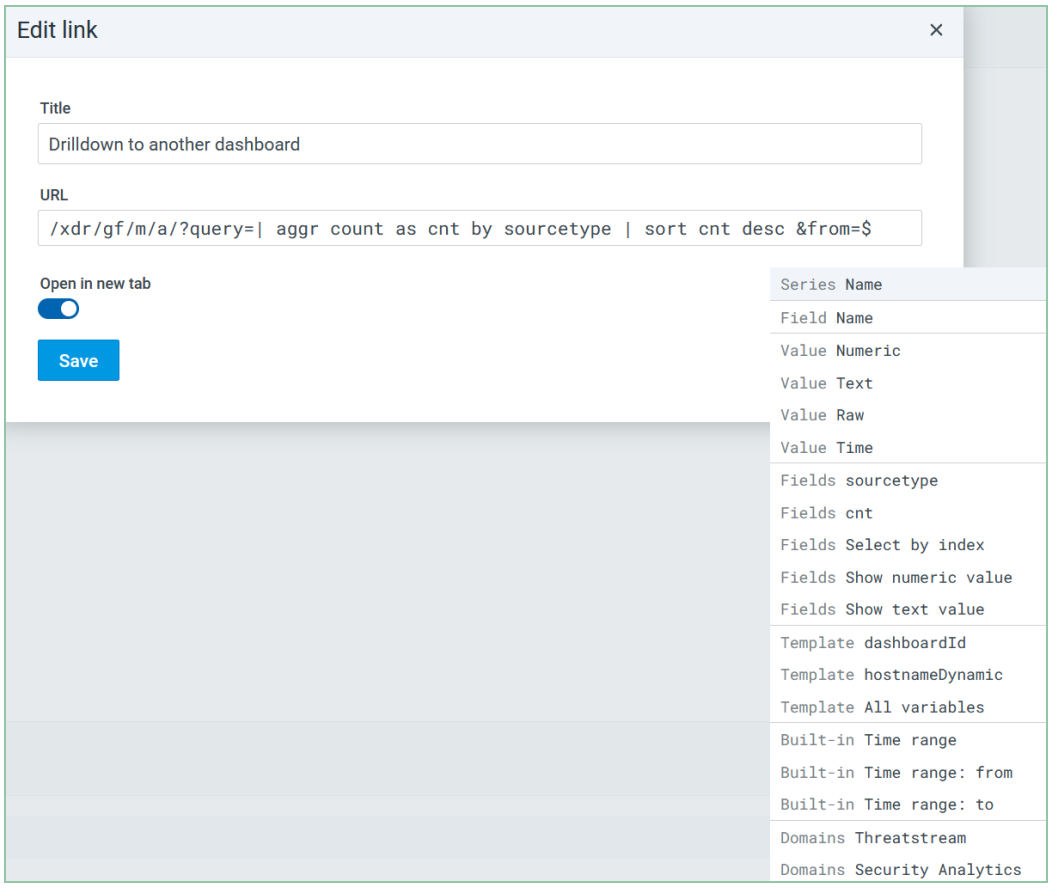
Use the options in the Panel tab to customize the options described in the following table.

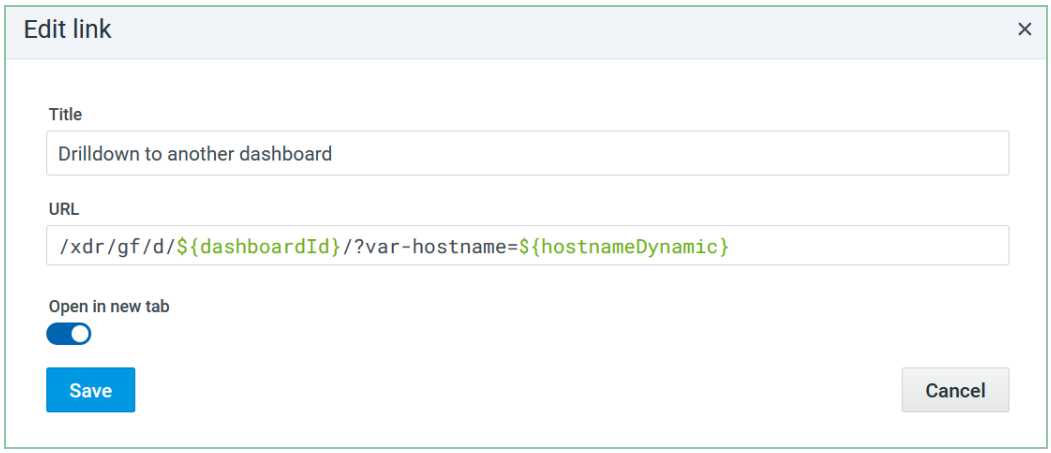
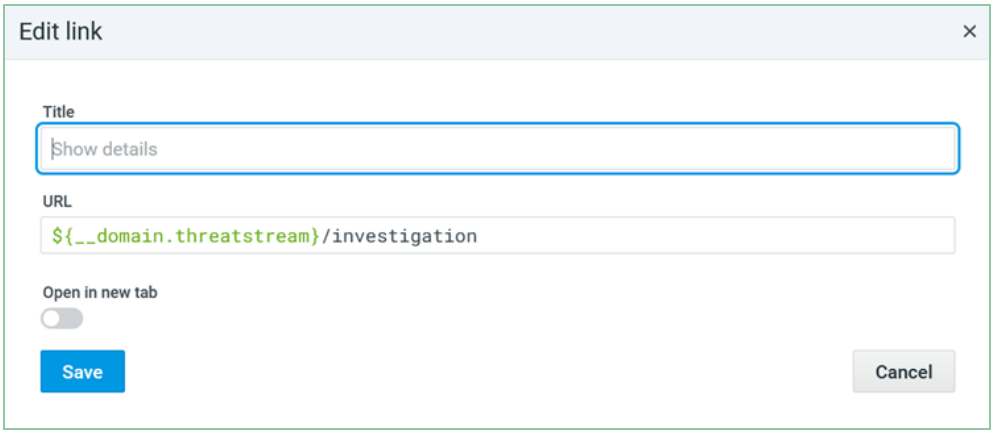
Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
Display	Scheme	Select a color scheme.
	Rotate x-axis tick labels	Specify the angle with which the labels will be rotated for better label readability.

Field Options

Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

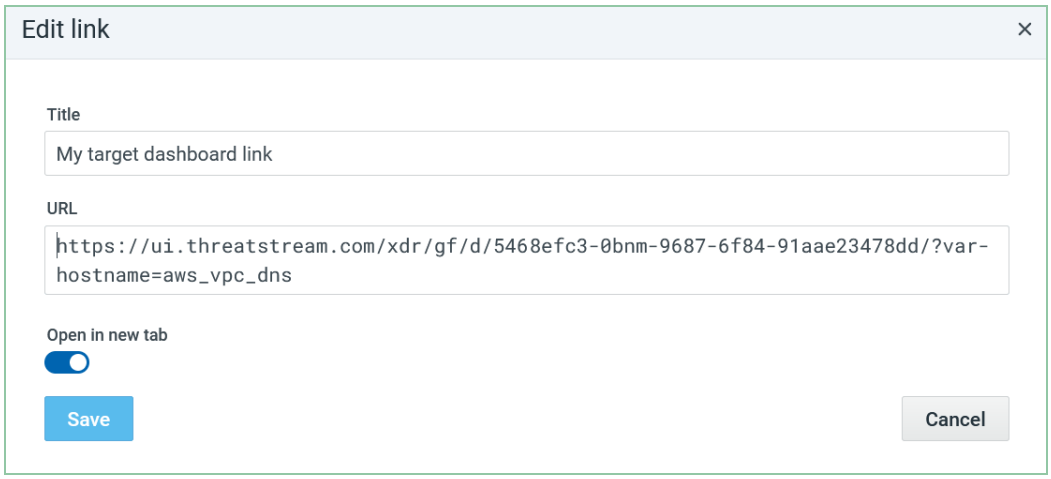
Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign ($) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("/"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**. 

Setting		Description
		In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream. **Add link to another dashboard** To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>`

Setting		Description
		Multiple variables: Multiple context variables and their corresponding values, separated by ampersand (&).

```
<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>
```



In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.

Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:

Setting		Description
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

About the Gauge Visualization Type

Use the Gauge visualization type to display a single statistical value.

AQL Expression Requirements

The gauge visualization type is used to show counts. The query must have the following elements to return the required data:

- A result set having a numeric column.
- Use the `aggr` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

```
| where category contains "Cloud" OR category contains "cloud"
| aggr count as cat_cnt by category
| sort cat_cnt desc
```

> **Note:** Using `addtotals=true` in your timechart operator query appends a total row and total column to your results. The `total` row shows the aggregate count per group across the entire time range, and the `total` column shows the aggregate count across all groups per time bucket. This is useful for comparing entities and spotting volume trends, without needing a separate query. See timechart operator for more information.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

Setting		Description
Display	Show	• Calculate • All values
	Limit	Max number of rows to display.
	Fields	Select the fields that should be included in the panel.
	Show threshold labels	Render the threshold values around the gauge bar.
	Show threshold markers	Renders the thresholds as an outer bar.
Text size	Title	Adjust the size of title text.
	Value	Adjust the size of value text.

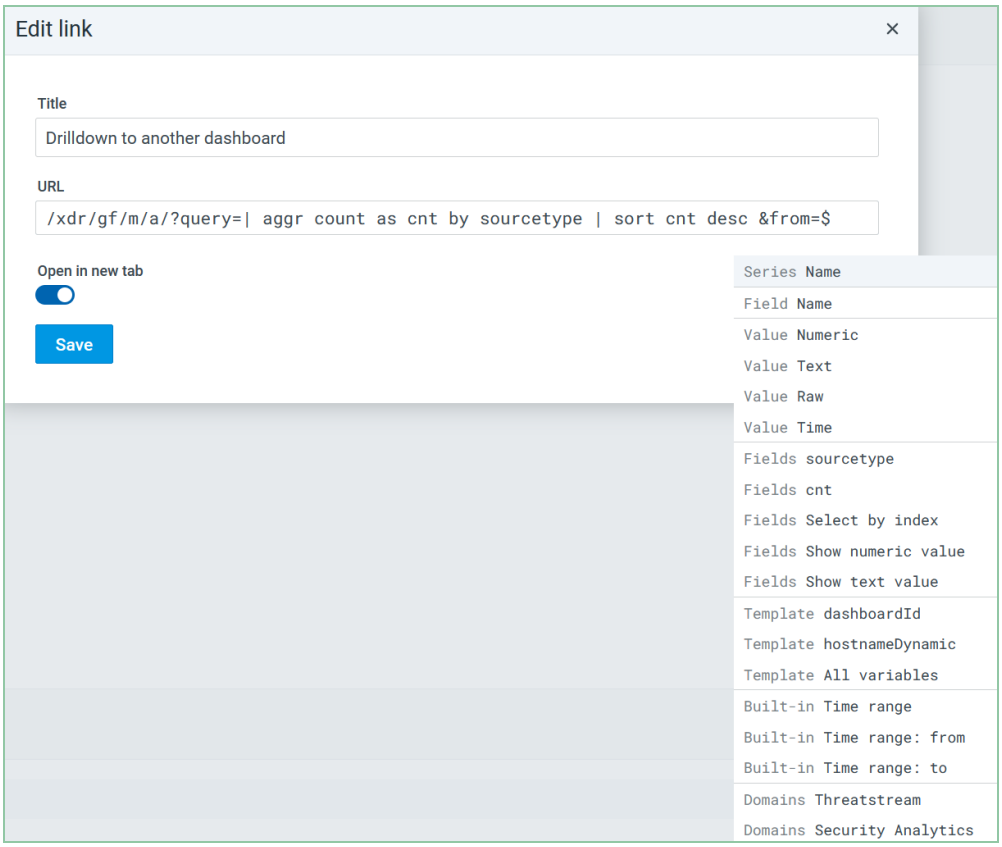
Field Options

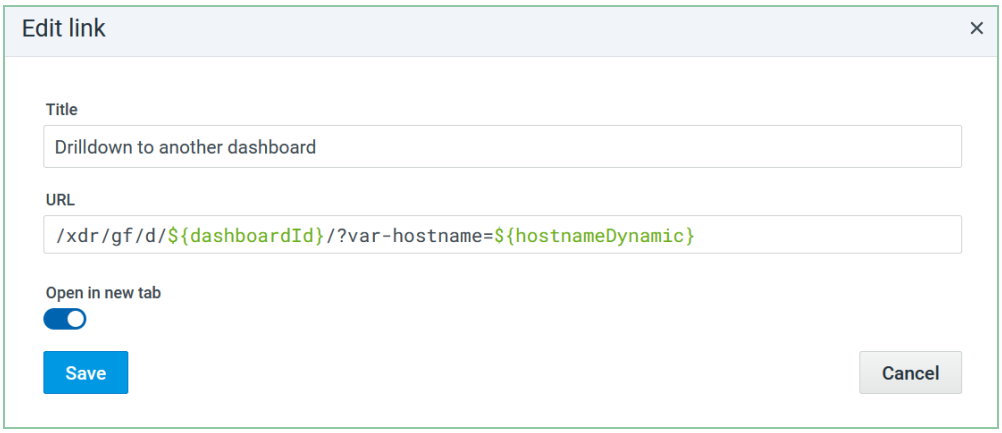
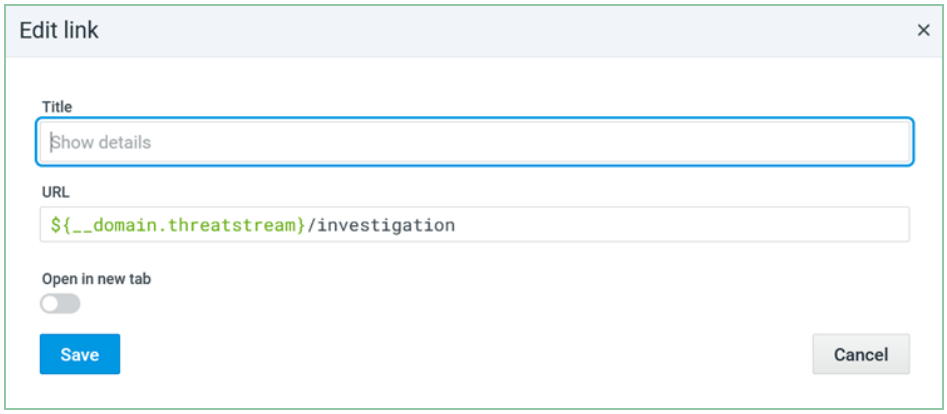
Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Standard options	Unit	Select a unit. Applies to all fields except time.
	Min	Specify a minimum value (as percentile). Leave empty to calculate based on all values.
	Max	Specify a maximum value (as percentile). Leave empty to calculate based on all values.
	Decimals	Number of decimal places in numeric values.
	Displayname	Change the field or series name.
	No value	What to show when there is no value.
	Colorscheme	Specify a single color or use the color palette.

Setting		Description
Thresholds	Add threshold	Configure mapping of thresholds to colors.
	Thresholds mode	• Absolute • Percentage
Value mappings	Add value mapping	Click **+Add value mapping** to map a single value or range of values to a specified text string. For example, map data having the country code `SG` to the text `Singapore`.

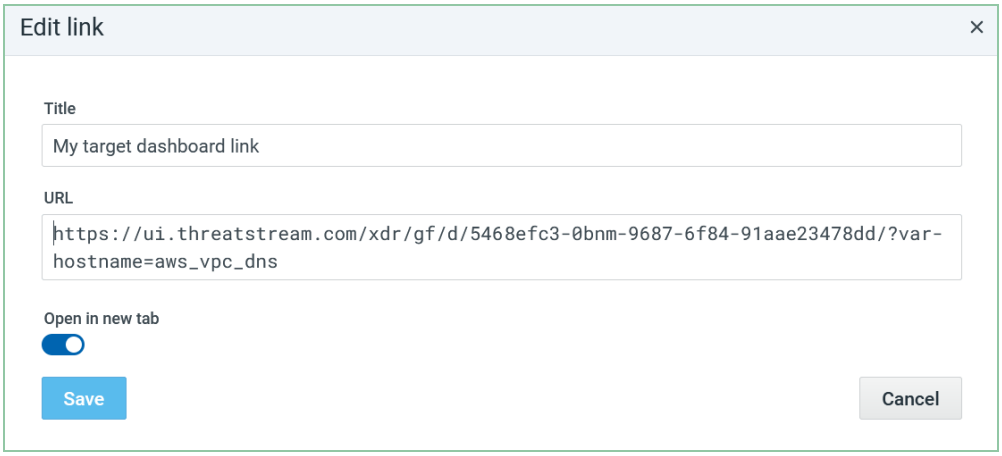
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

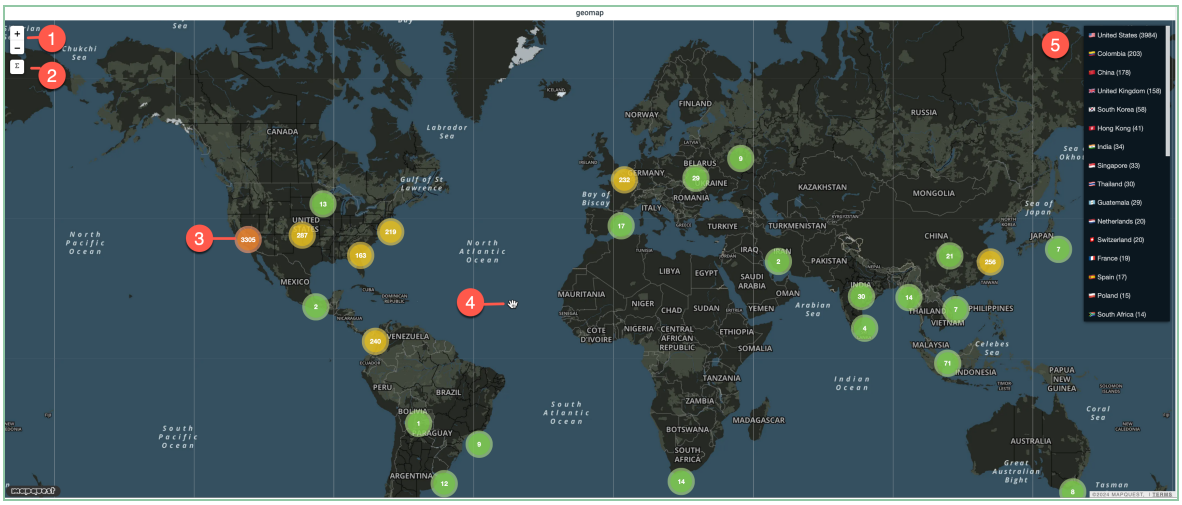
Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

About the Geo Map Visualization Type

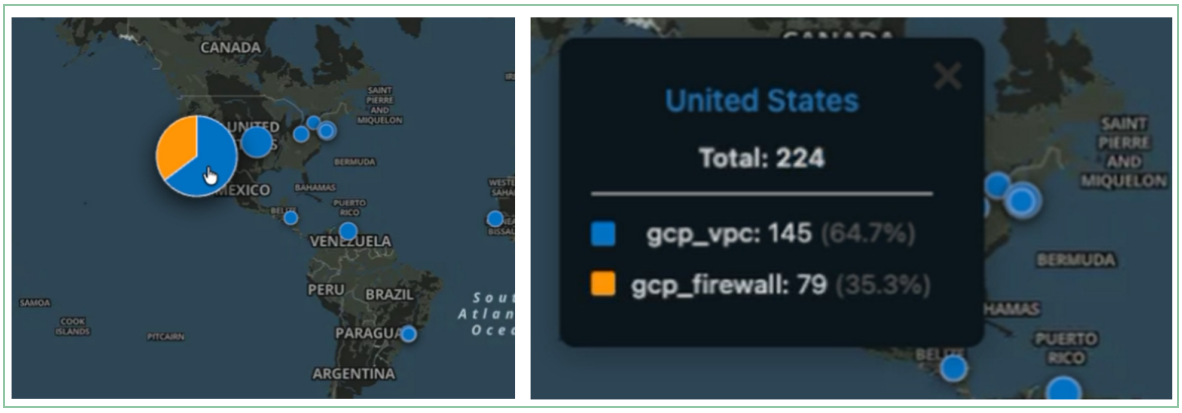
Use the Geo Map visualization type to show counts for a specified field displayed on a global map.



1	**Zoom**: Click + to Zoom in and - to zoom out.

2	**Aggregate by country**: • Click the ∑ icon to aggregate by country. For example, instead of displaying multiple latitude or longitude aggregations for the United States, you would see one count for the United States. • Click the ∑ icon when it is grayed to undo the aggregation by country.
3	**Count Bubble**: The size and color of the bubble differ according to count: • **0-99**: Small, Blue • **100-999**: Medium, Orange • **1000+**: Large, Red
4	**Drag map**: Click the map and use the hand ( ) pointer to drag the map as you like.
5	**Country list**: Counts are shown for each country. Click the in the country list to open a ThreatStream search filtered by country. When you click in the country list, the map also zooms into the location of the country on the map. Some countries have multiple IP locations. This gesture zooms to one of them.

Splitting by Categories



You can also split Geo Map bubbles by a categorical field. When you apply a split-by field, for example `sourcetype`, each country bubble becomes a pie chart that shows the relative contribution of each category to the total count at that location.

In the example above, the United States bubble is split by data source:

- The blue slice represents `gcp_vpc` events and the orange slice represents `gcp_firewall` events.
- When you click the bubble, a panel displays more information, showing the total count for the country along with the per-category counts and percentages.

This categorical splitting helps analysts compare different event outcomes or data sources at a quick glance, without leaving the map. Instead of building and managing multiple separate widgets, for example, one map for successful logins and another for failed logins, analysts can see both in a single bubble, with the pie slices showing how much each category contributes in each country.

AQL Expression Requirements

Geo Map requires specific data. The query must have the following elements to return the required data:

- Use the `iplocation` operator, which returns fields related to IP location.
- Aggregate results by the following location related fields: `country`, `latitude`, and `longitude`.

The following is an example of a query that returns the required data:

```
| where src_ip != "" and not (src_ip startswith "10." or src_ip
startswith "172."  or src_ip startswith "192" )  and action in
("allowed", "allow", "accept")| iplocation src_ip | aggr count as
cnt by country, lat, lon
```

Contact your Anomali Customer Success Manager if you have different requirements for Geo Map visualizations.

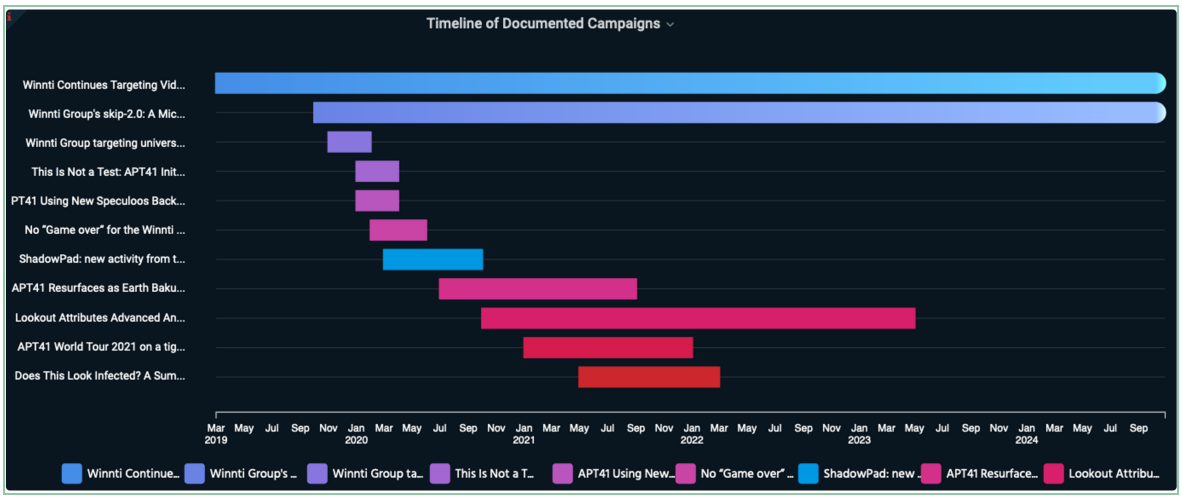
Panel Options

In Panel settings, you can customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

About the Gantt Time Series Visualization Type

Use the Gantt Time Series visualization type to analyze data based on date information, that provides visibility into intelligence and events across specific timeframes, and visualize these insights as a Gantt chart. For example, you can map campaigns by actors alongside the campaign start and end dates.



Use case

You can use the Gantt Time Series panels to analyze:

- User activity
- Event names
- Traffic flow over time
- File transfer events

AQL Expression Requirements

The Gantt Time Series visualization type can be used interactively to capture the timelines of multiple variables and expand them for sub-variable information, if any. The query must have the following elements to return the required data:

- The start date of the chart that populates its x-axis, for example, the `campaign_start_date`
- A field with names of the y-axis, such as, `campaign_names`

The following query is an example of a valid query that returns the required data:

```
threat_models
| where tm_type = "campaign"
| fields name as campaign_name, created_ts as campaign_start_date, modified_ts as campaign_end_date
| aggr count by campaign_name, campaign_start_date, campaign_end_date
```

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
Display	Grouping by	Optional. Variable by which you would like to group other variables that have shared attributes.
	Label Names	Variable that populates the Y-axis of the Gantt chart.
	Fields Start time	Variable that acts as the X-axis timeline of the Gantt chart to plot the progress of other variables of interest.
	Fields End time	Variable that signifies the end of the X-axis timeline of the Gantt chart.

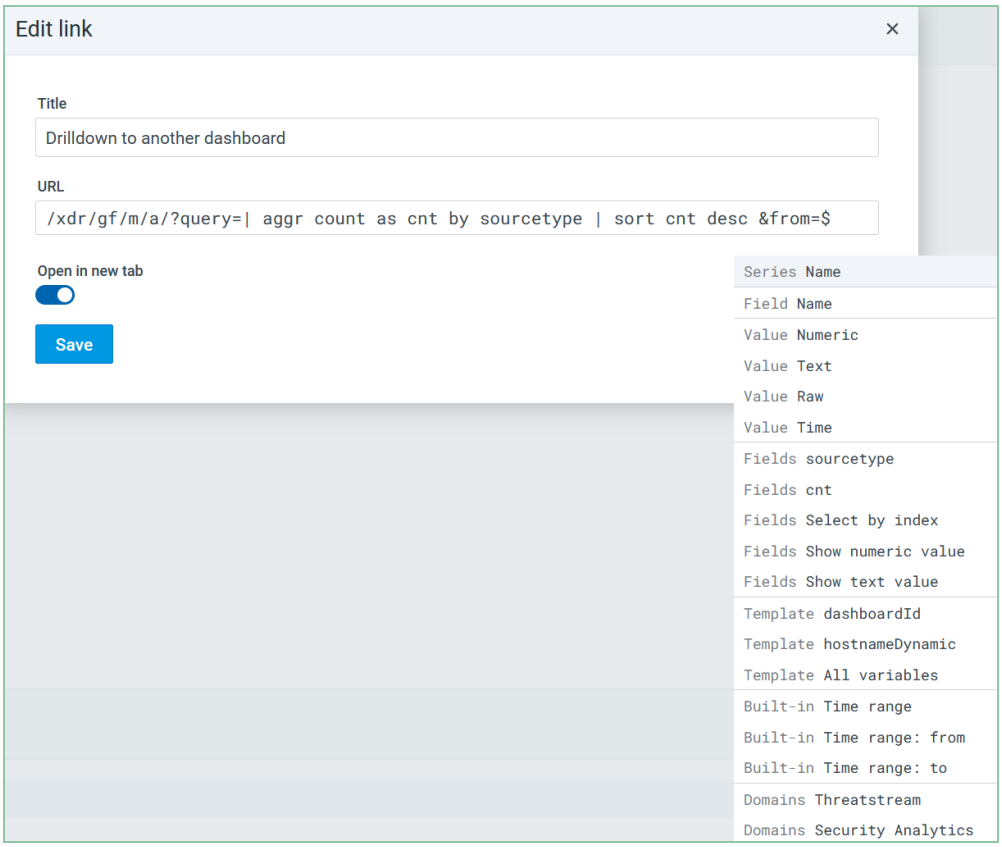
Field Options

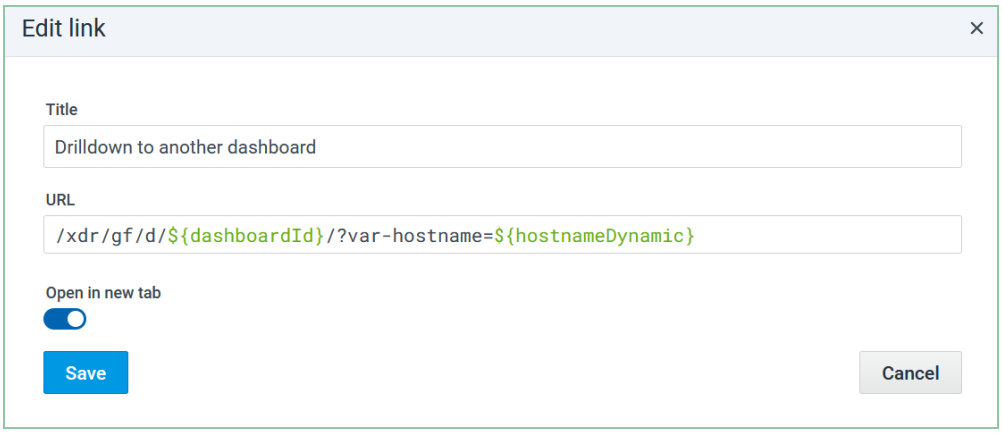
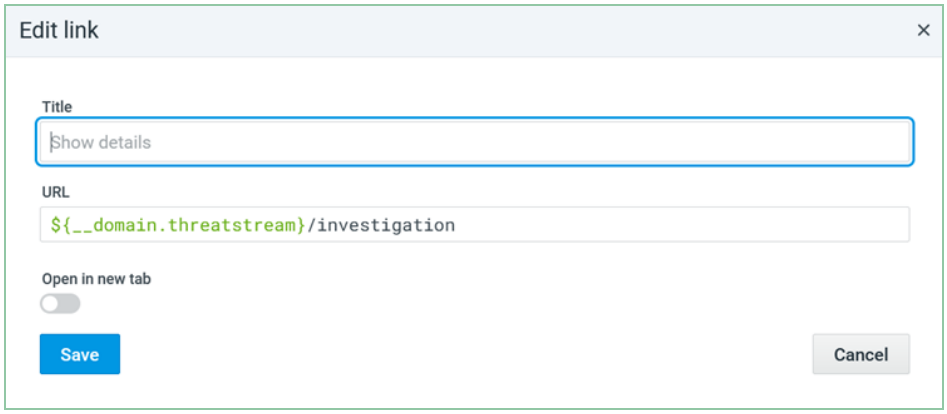
Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Standard options	Unit	Select a unit. Applies to all fields except time.
	Min	Specify a minimum value (as percentile). Leave empty to calculate based on all values.
	Max	Specify a maximum value (as percentile). Leave empty to calculate based on all values.
	Decimals	Number of decimal places in numeric values.
	Displayname	Change the field or series name.
	Novalue	What to show when there is no value.
	Colorscheme	Specify a single color or use the color palette.

Setting		Description
Thresholds	Add threshold	Configure mapping of thresholds to colors.
	Thresholds mode	• Absolute • Percentage
Value mappings	Add value mapping	Click **+Add value mapping** to map a single value or range of values to a specified text string. For example, map data having the country code `SG` to the text `Singapore`.

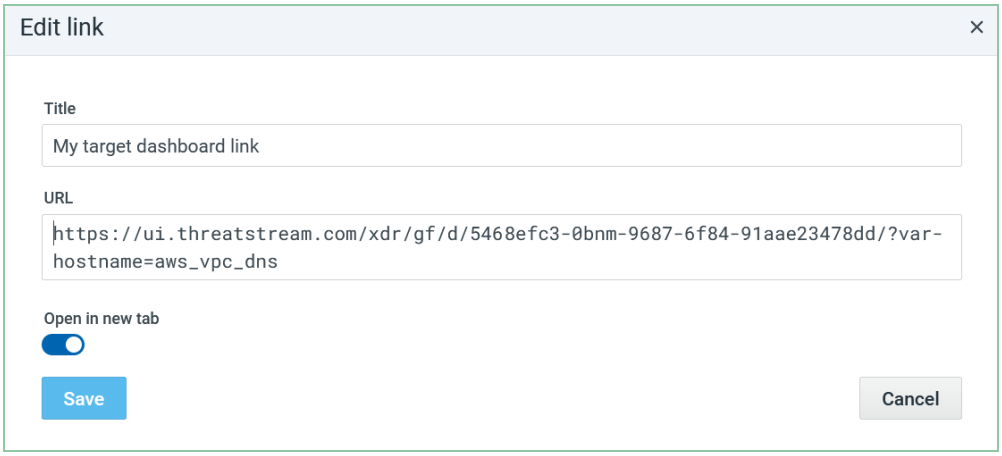
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

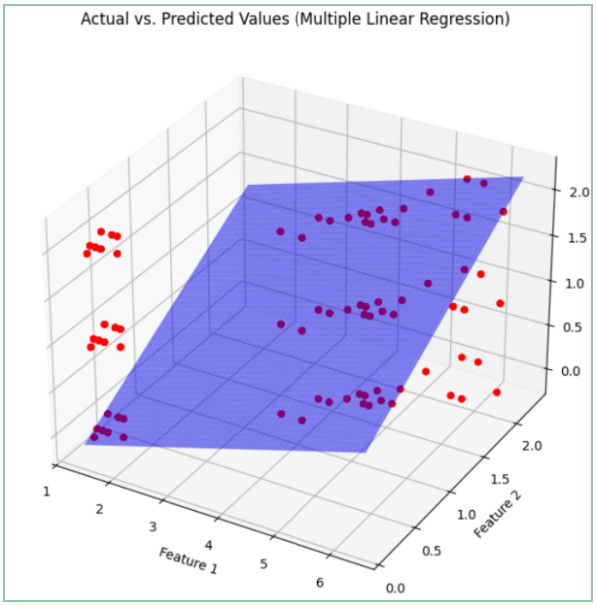
Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

About the Linear Regression Visualization Type

Use the Linear Regression visualization to explore how one or more independent variables relate to a dependent variable, and to evaluate the fit of a linear regression model. The visualization displays the original data as a scatter plot, with a fitted regression line for independent and dependent variables. It also includes optional residuals helping you assess how well the model fits your data.



Use case

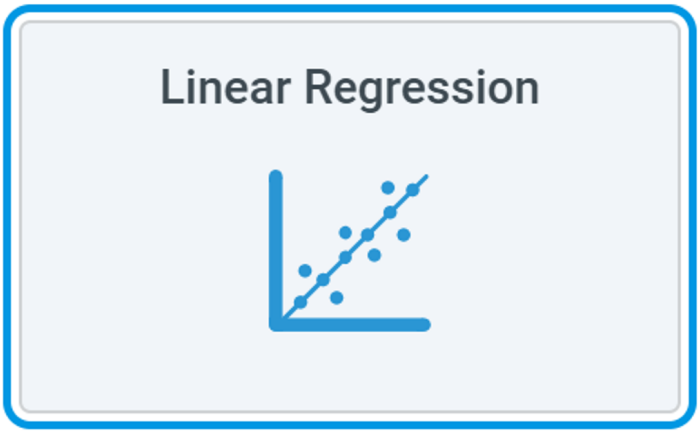
You can use the Linear Regression visualization to:

- analyze how a specific feature (for example, connection duration) correlates with an outcome (for example, likelihood of an alert firing), and visually inspect the fitted trend line.
- examine the combined effect of two variables, such as source IP reputation and destination port on a continuous outcome metric, using a regression plane.
- inspect residuals, that is the difference between predicted and actual values. to quickly spot non-linear patterns, outliers, or systematic bias in the model.
- compare multiple linear regression models by visually checking how tightly the data points cluster around the fitted line or surface.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

Setting		Description
Visualization	Linear Regression	Select the Linear Regression widget to create linear regression plots. 
Display	Show Model Info	Display a short message summarizing the model creation result above the chart.
	Enable Zoom	Allow users to zoom and pan within the chart for closer inspection of specific areas.
	Image Scale	Determines how the image should be scaled within the panel. Possible values are: Contain (fit within panel): Scale the chart to fully fit inside the panel while preserving its aspect ratio. Cover (fill panel): Scale the chart to fill the entire panel, which may crop edges to preserve aspect ratio. Auto (original size): Show the chart at its original size and let it scroll if it does not fit in the panel.

Examples

Example 1: Visualize single-variable linear regression

```
| fit OrdinalEncoder action src | fit LinearRegression encoded_
action from encoded_src into 'linear_regression_model'
```

Example 2: Visualize single-variable linear regression with residuals

```
| fit OrdinalEncoder action src | fit LinearRegression encoded_
action from encoded_src into 'linear_regression_model2' with_
residuals=true
```

Example 3: Visualize multiple linear regression

```
| fit OrdinalEncoder action src src_port | fit LinearRegression
encoded_action from encoded_src encoded_src_port into 'linear_
regression_model3'
```

About the MITRE ATT&CK Flow Visualization Type

Use the MITRE ATT&CK Flow visualization type to create a dashboard panel that displays an attack flow. This visualization generates an attack flow from your AQL search results or an imported JSON or Excel file. You can then export the attack flow from the dashboard in JSON format.

AQL Expression Requirements

The MITRE Attack Flow visualization type is used to display attack flows from your Attack Flow Library on ThreatStream. See Attack Flow Library for details.

To create an attack flow on the Visualization tab of Search results, run an AQL query in Search using the "superapi" on page 666 operator and the attack flow endpoint in the following format:

```
anomali_api_proxy | superapi

endpoint = "/api/v1/attackflow/<attack_flow_id>"

method = get
```

Below is an example of an attack flow generated with the help of the MITRE Attack Flow visualization type.

Panel Options

Use settings in the Panel tab to customize the options described in the following table:

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

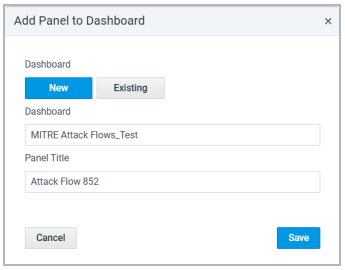
Adding a MITRE ATTACK Flow Panel to a Dashboard

You can add a MITRE ATT&CK Flow panel to a dashboard from the Visualization tab or directly from a dashboard.

To add a MITRE ATT&CK Flow panel to a dashboard from the Visualization tab:

1. On the Visualization tab, click **Add to Dashboard**.
2. In the **Add Panel to Dashboard** dialog box that opens:

- To add the panel to a new dashboard, enter a dashboard name and a panel name.



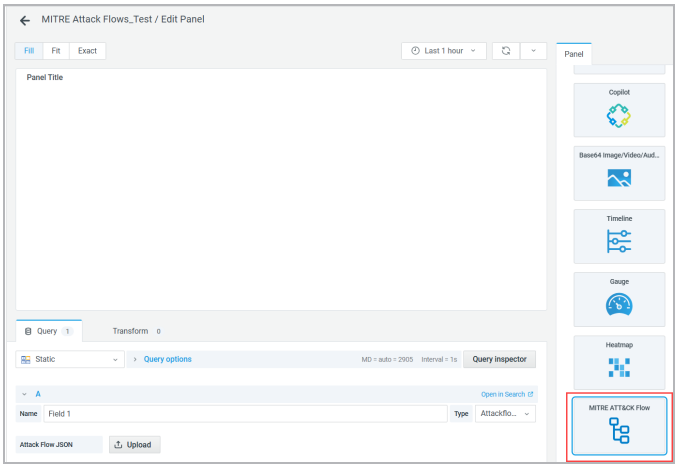
- To add the panel to an existing dashboard, click **Existing**, select the dashboard to which you want to add the panel, and enter a panel name.

3. Click **Save**.

   The panel is added to the dashboard.

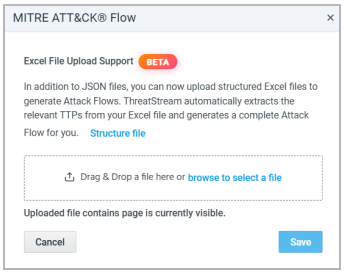
To add a MITRE ATT&CK Flow panel from a dashboard:

1. Navigate to **Dashboard** > **Library**.
2. Select the dashboard to which you want to add a MITRE ATT&CK Flow panel.
3. Click **Edit**.
4. Click **Add an empty panel**.
5. On the Edit Panel page, select the MITRE ATT&CK Flow visualization type.



6. Click **Upload**.
7. Drag & drop or select an attack flow file in JSON or Excel format.

> **Note:** Only MITRE attack flows v2.0 can be imported as JSON files.



8. Click **Save**. The attack flow appears on the panel.
9. Click **Save** on the dashboard to save the newly created MITRE ATT&CK Flow panel.

Exporting Attack Flows

Attack flows displayed on the Visualization tab or dashboard panels can be exported in JSON format.

To export an attack flow, click the **Export** icon ( ) in the bottom right corner.

About the MITRE ATT&CK Heatmap Visualization Type

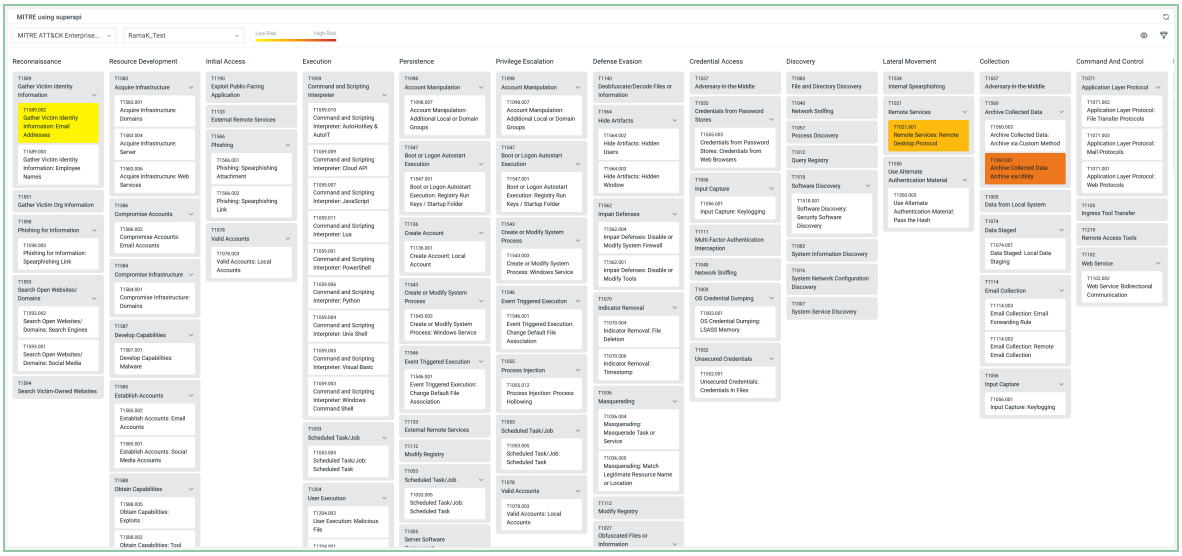
Use the MITRE ATT&CK Heatmap visualization to view adversarial behaviors in your organization mapped to the MITRE ATT&CK matrix based on ThreatStream threat models.

The visualization helps you capture different tactics, techniques, procedures (TTPs), including sub-techniques, from your Anomali Query Language (AQL) search results and uses ThreatStream to display your MITRE ATT&CK Profiles in your organization.

This visualization also supports three complementary MITRE view types:

- **Security Controls** to visualize control coverage and gaps
- **Threat Model** to focus on techniques tied to your modeled threats
- **Frequency** to highlight the techniques most frequently observed in your AQL results.

See "MITRE View Type" on page 319 for more information.



AQL Expression Requirements

You can use AQL queries to create a MITRE ATT&CK heatmap and view them on the **Visualization** tab of the Search results page.

For example, the following AQL query uses the "superapi" on page 666 operator and the ThreatStream `actor` endpoint to fetch up to 1000 attack patterns for the actor `123456` and display their ID, name, and MITRE attribution status.

```
example_test_table

| superapi
endpoint='/api/v1/actor/123456/attackpattern/?limit=1000&offset=
0' output_types='string' method=GET

| calc objects=json_extract(response_data, 'objects', ARRAY)

| mvexpand objects
```

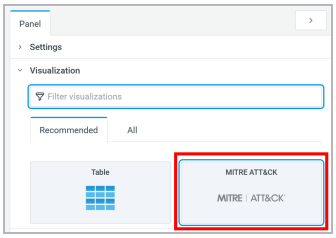
```
| calc name=json_extract(objects_expanded, 'name')

| calc id=json_extract(objects_expanded, 'id')

| calc is_mitre=json_extract(objects_expanded,  'is_mitre')

| fields id, is_mitre, name
```

In the following example, the **Frequency** MITRE view type aggregates MITRE technique frequency counts from threat intelligence reports based on MITRE technique ID and report count.

```
threat_models

| where tm_type = "tipreport"

| where feed_id = 6634 or feed_id = 382 or feed_id = 3404 or
feed_id = 6853

| mvexpand tags

| where tags_expanded startswith "GIR:"

| lookup intel_GIR_lookup on tags_expanded=gir_code output
description as gir_description

| lookup gir_to_mitre_lookup on tags_expanded=gir_code output
mitre_tactic as mitre_tactic1 , mitre_technique_id, mitre_
technique_name

| where mitre_technique_id != ""

| aggr dc(id) as report_count by mitre_technique_id

| sort report_count desc
```

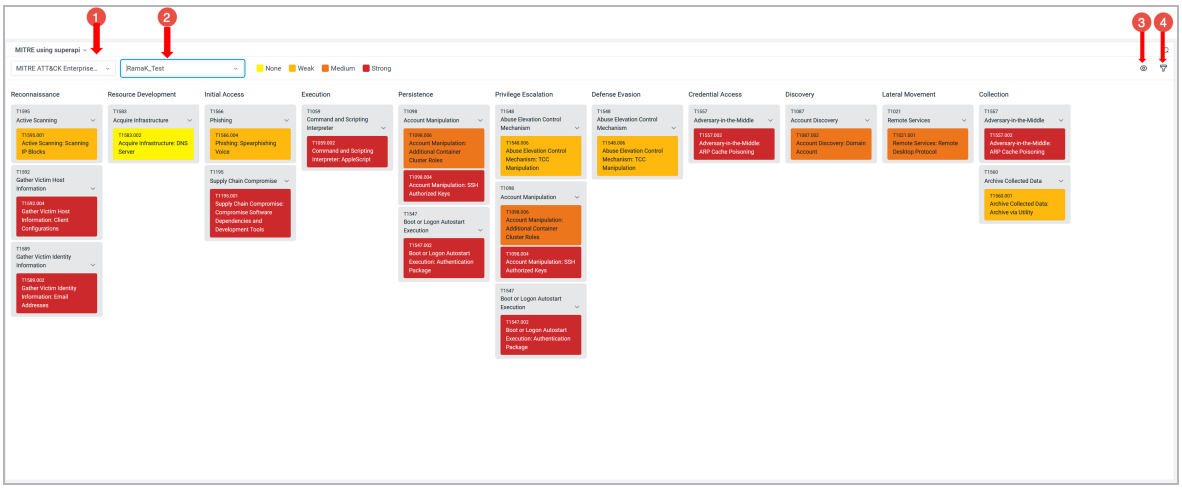
Panel Options

The **Panel** tab lets you customize the following visualization options:

Setting	Fields	Description
Settings	**Panel title**	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
	Show Description	Display the panel description.
Visualization	**All visualizations**	Select the **MITRE ATT&CK** visualization from the list of visualizations. 
Display	**MITRE View Type**	Select the MITRE view type. Possible values are: • **Security Controls** • **Threat Model** • **Frequency** See "MITRE View Type" below for more information.

MITRE View Type

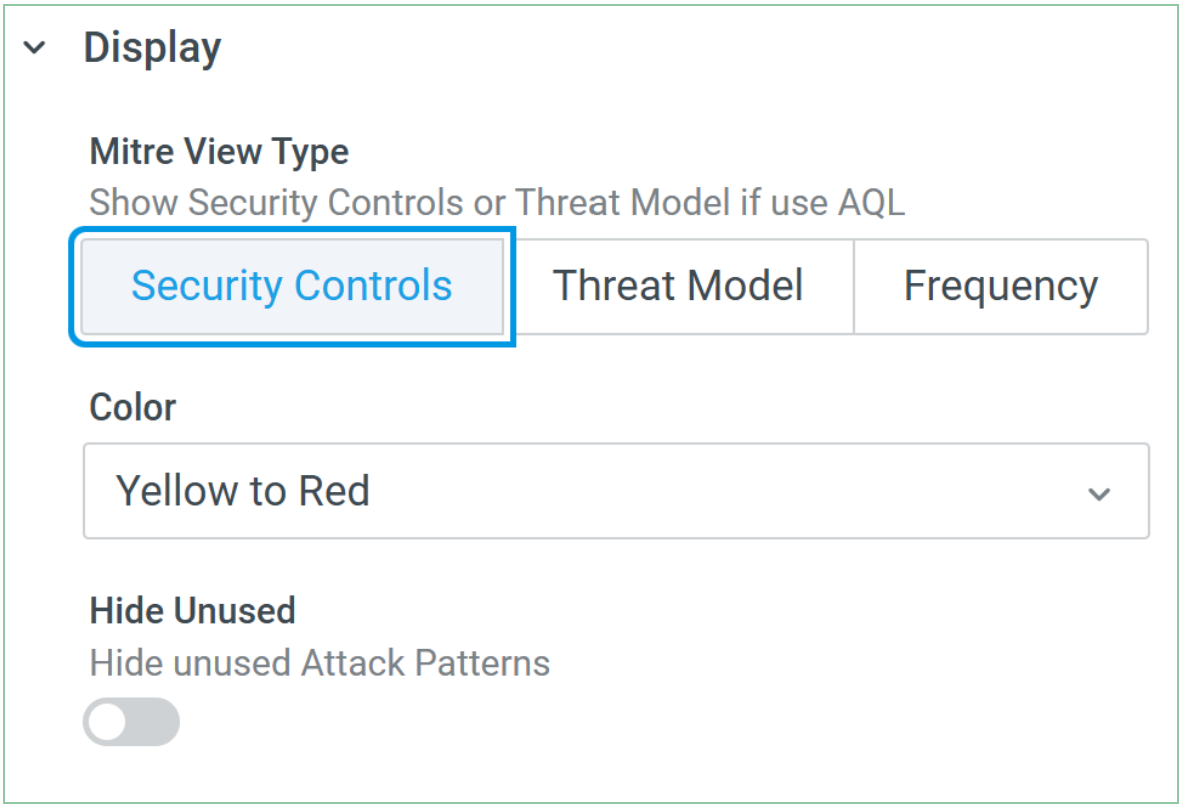
The following settings are common to the **Security Controls**, **Threat Model**, and **Frequency** MITRE View Types.



1	**Current MITRE ATT&CK framework Type and Version:** Shows the default MITRE ATT&CK framework version used by your organization. See Specifying a MITRE ATT&CK Security Coverage Framework Version for more information on setting a specific version. Depending on the MITRE ATT&CK version, you can select one of the following MITRE ATT&CK framework types: • **Enterprise**: Shows a comprehensive matrix of tactics and techniques applied against enterprise infrastructures. Available for all supported MITRE ATT&CK framework versions. • **Mobile**: Shows a detailed matrix of techniques involving Android and iOS device access and network-based effects that can be used by attackers without device access. Available for MITRE ATT&CK Version 7.X and later. • **ICS**: Shows how attackers achieve a tactical goal by performing an action. Available for MITRE ATT&CK Version 8.X and later. See the ThreatStream Using the MITRE ATT&CK Framework in Investigations and Using MITRE ATT&CK Frameworks in Threat Models guides for more information on MITRE investigations and threat models.

2	**Current MITRE ATT&CK Profile:** Shows the MITRE ATT&CK profile selected to represent a security coverage.
3	**Show or Hide Sub-techniques**: If your organization has configured an applicable MITRE version, click the eye icon ( ) to show or hide sub-techniques corresponding to a technique. **Note:** Sub-techniques are visible in the matrix for MITRE ATT&CK Enterprise Version 7.2 and later.
4	**Filter**: Filter techniques by **Platform** and **Entities**. **Note:** Filters are applicable only to MITRE ATT&CK Version 6.2. Filter techniques are deprecated in MITRE ATT&CK Version 7.2 and later.

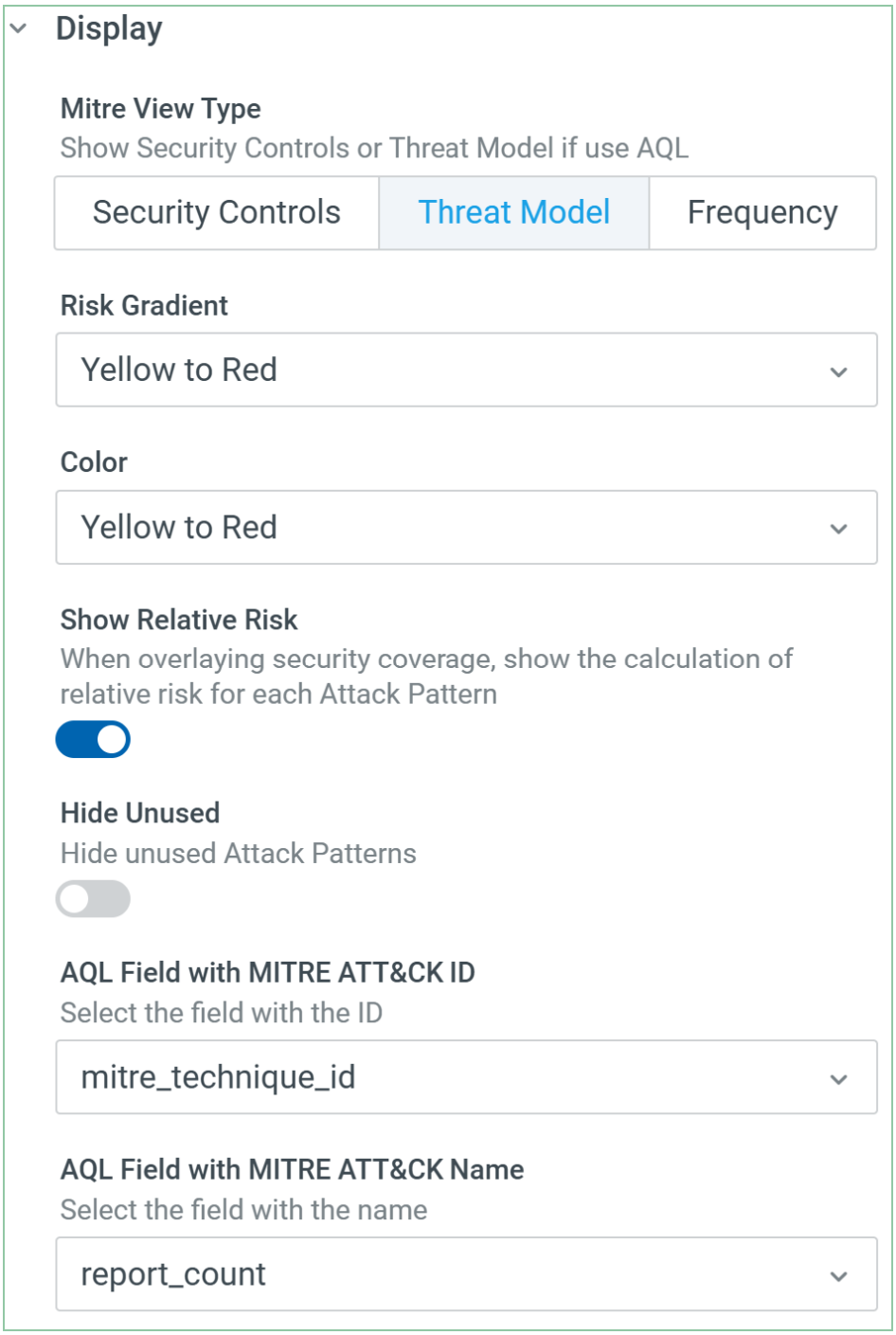
Security Controls



Configure the following **Panel** settings for the **Security Controls** MITRE view type:

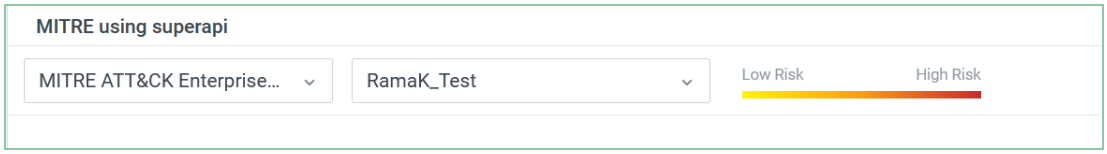
- In the **Panel** tab, click **Edit** and select the color gradient from the **Color** list. This setting defines the color gradient for the MITRE profile security coverage matrix.
  - In the example screenshot above, the gradient ranges from Yellow to Red.
- **Hide Unused Attack Patterns**: Toggle the **Hide Unused** switch to display only those TTPs for which you have configured a security coverage level.

Threat Model



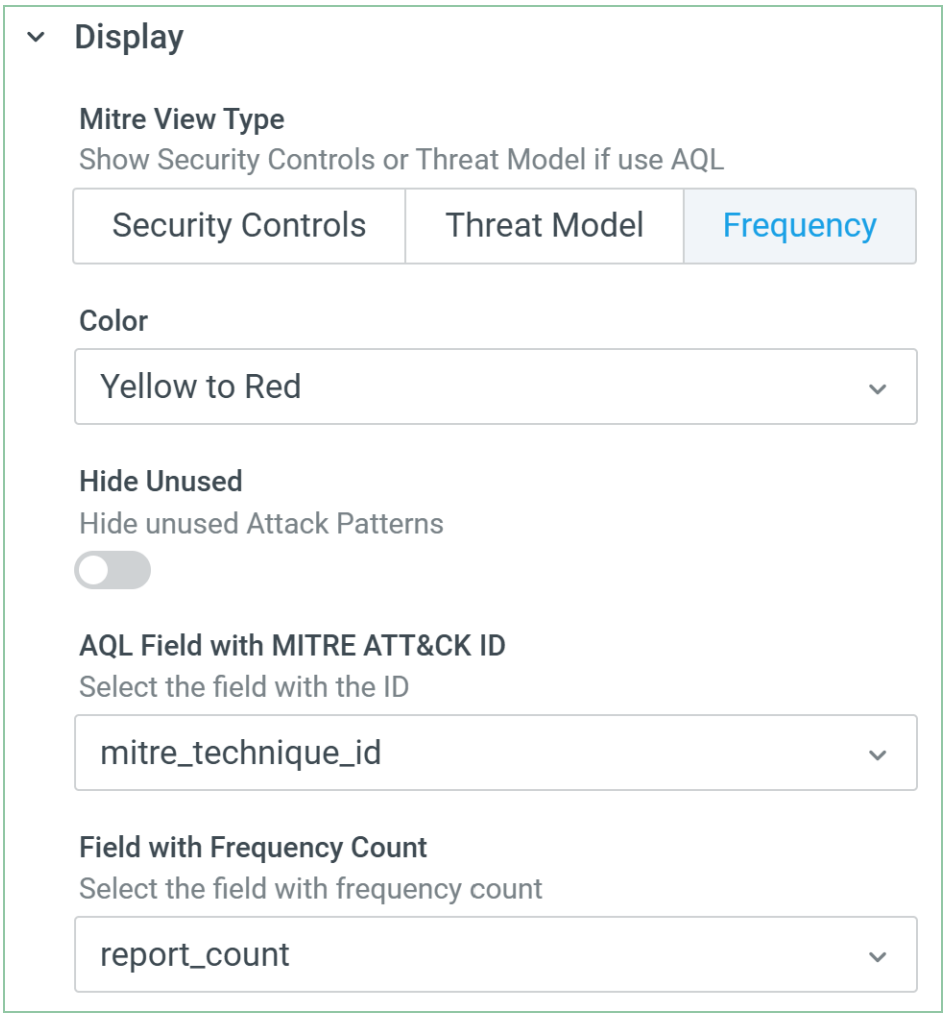
Configure the following **Panel** settings for the **Threat Model** MITRE view type:

- In the **Panel** tab, click **Edit** and select the color gradient of the profile risk from the **Risk Gradient** list. This becomes the color gradient for your MITRE profile security coverage matrix.
  - In the example screenshot below, the risk gradient ranges from Low Risk (Yellow) to High Risk (Red).



    - You can also enable the **Show Relative Risk** switch to show the calculations of relative risk for each attack pattern.
- **Hide Unused Attack Patterns**: Enable the **Hide Unused** switch to display only those TTPs for which you have configured a security coverage level.
- **AQL Field with MITRE ATT&CK ID**: Select the field with the ID (from your AQL query) as the MITRE ATT&CK ID.
- **AQL Field with MITRE ATT&CK Name**: Select the field with the name (from your AQL query) as the MITRE ATT&CK name.

Frequency



Configure the following **Panel** settings for the **Frequency** MITRE view type:

- **Color**

  Select the color gradient that represents the intensity of activity for each MITRE ATT&CK technique. This gradient is applied to the matrix based on the frequency count returned by your AQL query.

  For example, a Yellow to Red gradient displays less frequent techniques in lighter colors and more frequent techniques in darker colors.

- **Hide Unused Attack Patterns**

Toggle the **Hide Unused** switch to display only those TTPs for which your AQL query returns at least one occurrence. Techniques that do not appear in the AQL results are hidden from the matrix.

- **AQL Field with MITRE ATT&CK ID**

  Select the field in your AQL results that contains the MITRE ATT&CK technique or sub-technique ID (for example, `mitre_technique_id`).

  This field is used to map each row in your AQL result set to a specific technique on the MITRE ATT&CK matrix.

- **Field with Frequency Count**

  Select the numeric field in your AQL results that contains the frequency or count value (for example, `report_count`).

  The heatmap uses this value to color each technique cell. The color gradient automatically scales between the minimum non-zero value and the maximum frequency value in the results. This way, the minimum value maps to the lightest end of the gradient, the maximum value maps to the darkest end, and any value in between is colored proportionally between this range, all without any manual computation.

Adding a Heatmap to a Dashboard

You can add a MITRE ATT&CK Heatmap panel to a dashboard from either the **Visualization** tab or directly from a dashboard.

To add the panel from the Visualization tab:

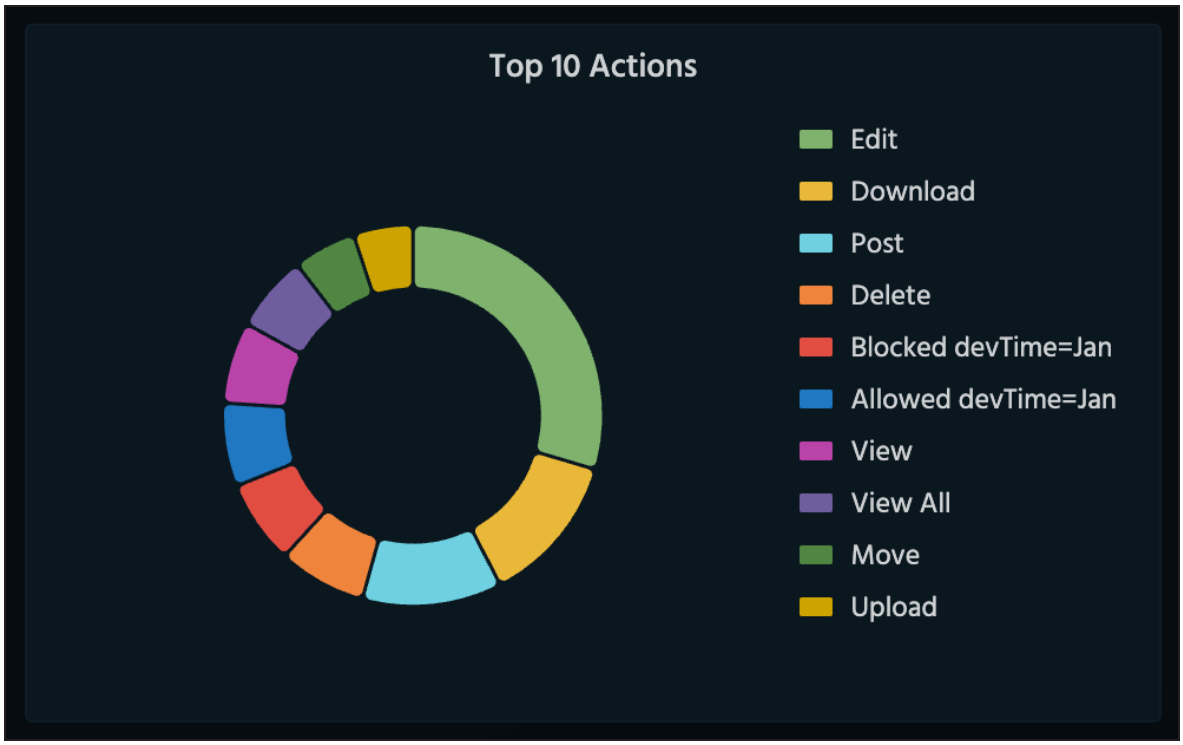
1. Click **Add to Dashboard**.
2. In the **Add Panel to Dashboard** dialog box:
   - To add the panel to a new dashboard, enter a dashboard name and a panel name.
   - To add the panel to an existing dashboard, click **Existing**, select the desired dashboard, and enter a panel name.
3. Click **Save** to add the panel to the dashboard.

To add the panel from a dashboard:

1. Navigate to **Dashboard** > **Library**.
2. Select the dashboard to which you want to add the panel.
3. Click **Edit**.
4. Click > **Add an empty panel**.
5. On the **Edit Panel** page, in the **Panel** tab, select the **MITRE ATT&CK Heatmap** visualization.
6. Click **Save** to add the heatmap to the panel.
7. Click **Save** on the dashboard to save the newly created MITRE ATT&CK Heatmap in the dashboard.

About the Pie Chart Visualization Type

Use the Pie Chart visualization type to show aggregate counts of values for a single field.



AQL Expression Requirements

The pie chart visualization type is used to show counts. The query must have the following elements to return the required data:

- A result set having a string column and a numeric column.
- Use the `aggr` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

```
| aggr count as cnt by sourcetype |sort cnt desc | limit 10
```

> **Note:** Using `addtotals=true` in your timechart operator query appends a total row and total column to your results. The `total` row shows the aggregate count per group across the entire time range, and the `total` column shows the aggregate count across all groups per time bucket. This is useful for comparing entities and spotting volume trends, without needing a separate query. See timechart operator for more information.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

Setting		Description
Display	Show	• Calculate • All values
	Limit	Max number of rows to display.
	Fields	Select the fields that should be included in the panel.
	Piechart type	• Pie • Donut
	Labels	Select the labels to be displayed in the pie chart.
	Legend mode	• List • Table • Hidden
	Legend placement	• Bottom • Right
	Legend calculations	Select functions. Advanced users only.
Text size	Title	Adjust the size of title text.
	Value	Adjust the size of value text.

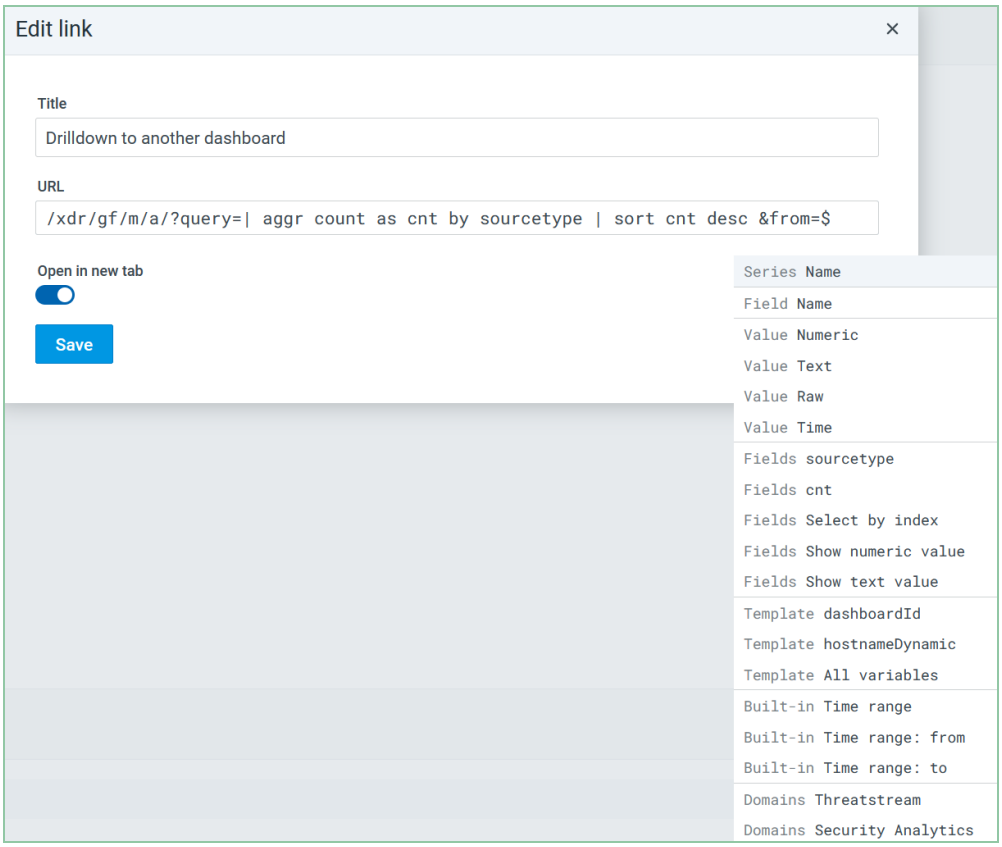
Field Options

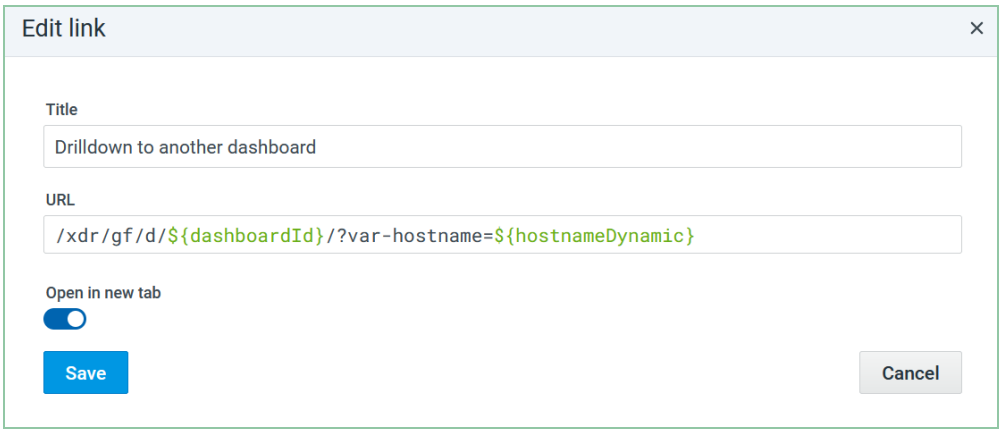
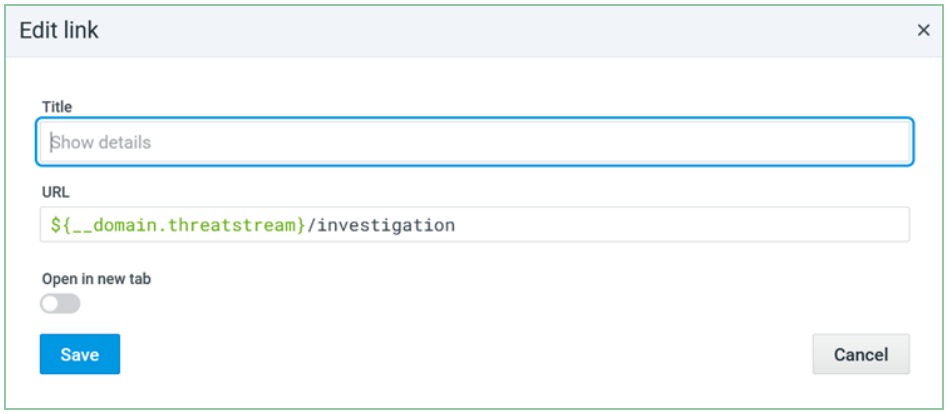
Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Standard options	Unit	Select a unit. Applies to all fields except time.
	Min	Specify a minimum value (as percentile). Leave empty to calculate based on all values.
	Max	Specify a maximum value (as percentile). Leave empty to calculate based on all values.
	Decimals	Number of decimal places in numeric values.
	Displayname	Change the field or series name.
	Novalue	What to show when there is no value.
	Colorscheme	Specify a single color or use the color palette.

Setting		Description
Thresholds	Add threshold	Configure mapping of thresholds to colors.
	Thresholds mode	• Absolute • Percentage
Value mappings	Add value mapping	Click **+Add value mapping** to map a single value or range of values to a specified text string. For example, map data having the country code `SG` to the text `Singapore`.

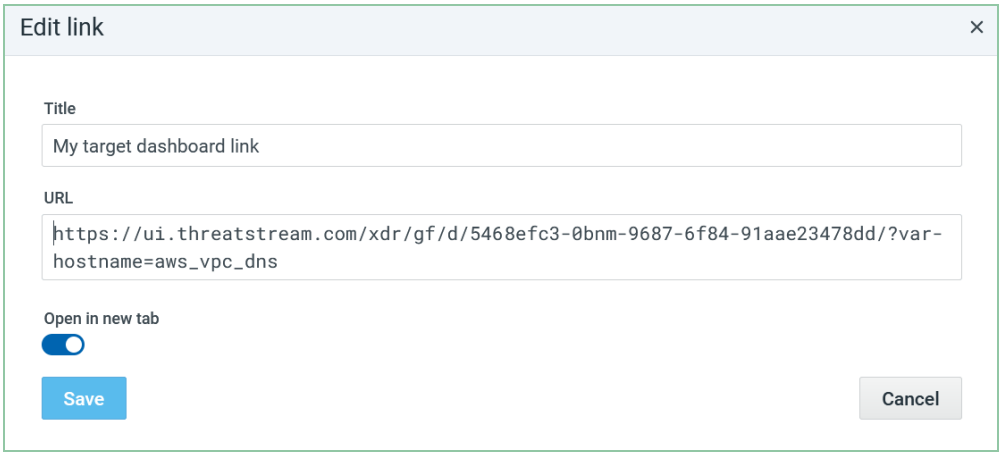
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

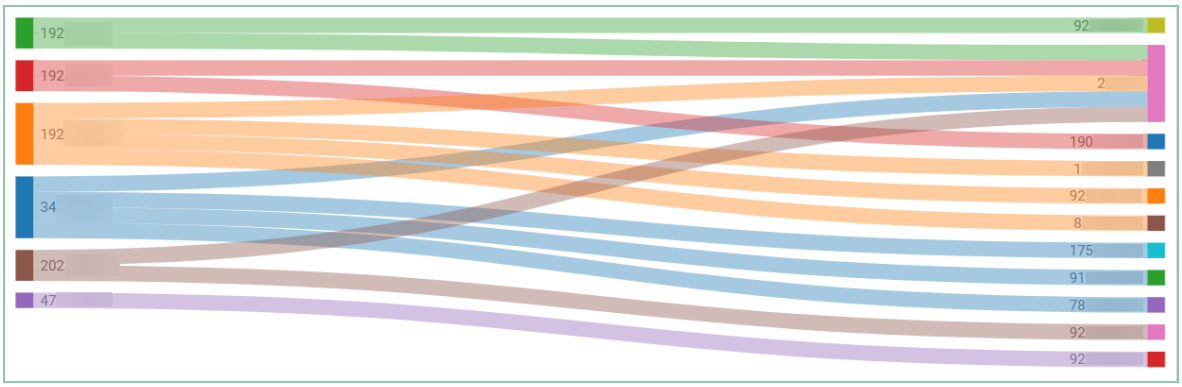
Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

About the Sankey Diagram Visualization Type

Use the Sankey Diagram visualization type to display flows where the width of the lines is proportional to the flow rate.



Sankey diagrams can be effective for visualizing how resources, data, or processes are distributed or moving between different stages or categories.

You can use Sankey diagrams to illustrate the flow of data, the relationships between systems or events, and to understand attack paths or the movement of threats within a network. For example:

- To visualize the flow of network traffic between different segments or devices.

  A Sankey diagram can show the volume of traffic moving between different network zones, such as from external sources to the DMZ and then to the internal

network. This can help identify unusual traffic patterns that might indicate lateral movement or data exfiltration.

- To correlate events across different sources and understand the relationships between them.

  A Sankey diagram can illustrate how different log sources contribute to a correlated security incident, showing the flow of events from initial detection in a firewall to correlation and then the alert generation.

- To understand how an attacker might move through the network from initial compromise to their end goal.

  A diagram can show an attacker's journey from phishing to initial compromise, privilege escalation, lateral movement, and finally, data exfiltration. This visualization can help in comprehending the complexity and potential impact of an attack.

AQL Expression Requirements

The Sankey diagram visualization type is used to show many-to-many connections between values of two columns. The width of each connection is proportionate to counts in each pair of values.

The query must have the following elements to return the required data:

- At least 2 fields that will serve as the source and target of the flow.
- 1 field that will serve as the flow value (for example, volume of traffic).
- Use the `aggr` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

```
eventlog | where src is not null and src!="" and dest is not null
and dest!='N/A' and bytes_in>0
|aggr sum(bytes_in) as bytes, count as cnt by src, dest
|sort cnt desc
```

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

About the Stat Visualization Type

Use the Stat visualization type to display a single statistical value.

AQL Expression Requirements

The stat visualization type is used to show counts. The query must have the following elements to return the required data:

- A result set having a numeric column.
- Use the `aggr` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

```
| aggr count as cat_cnt by category |sort cat_cnt desc | limit 5
```

Note: Using `addtotals=true` in your timechart operator query appends a total row and total column to your results. The `total` row shows the aggregate count per group across the entire time range, and the `total` column shows the aggregate count across all groups per time bucket. This is useful for comparing entities and spotting volume trends, without needing a separate query. See timechart operator for more information.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

Setting		Description
Display	Show	• Calculate • All values
	Limit	Max number of rows to display.
	Fields	Select the fields that should be included in the panel.
	Orientation	Stacking direction in case of multiple series or fields. • Auto • Horizontal • Vertical
	Text mode	Control if name and value is displayed or just name.
	Color mode	Color either the value or the background. • Value • Background
	Graph mode	Stat panel graph / sparkline mode. • None • Area
	Alignment mode	Value and title positioning. • Auto • Center
Text size	Title	Adjust the size of title text.
	Value	Adjust the size of value text.

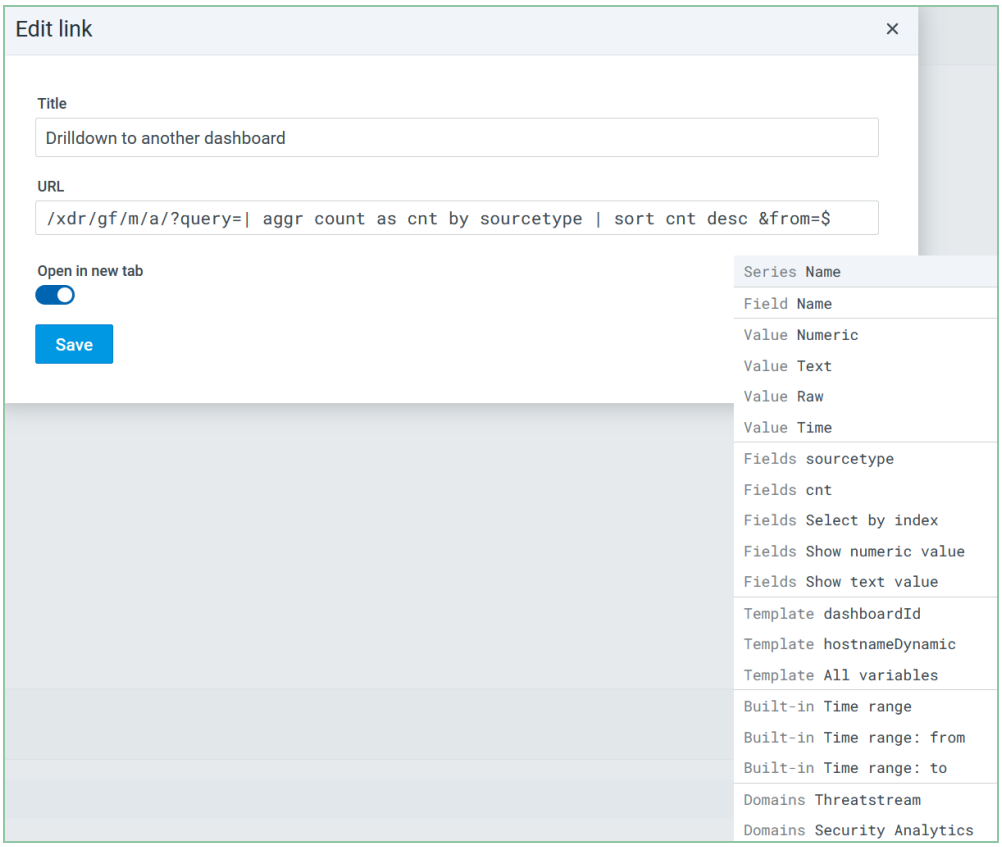
Field Options

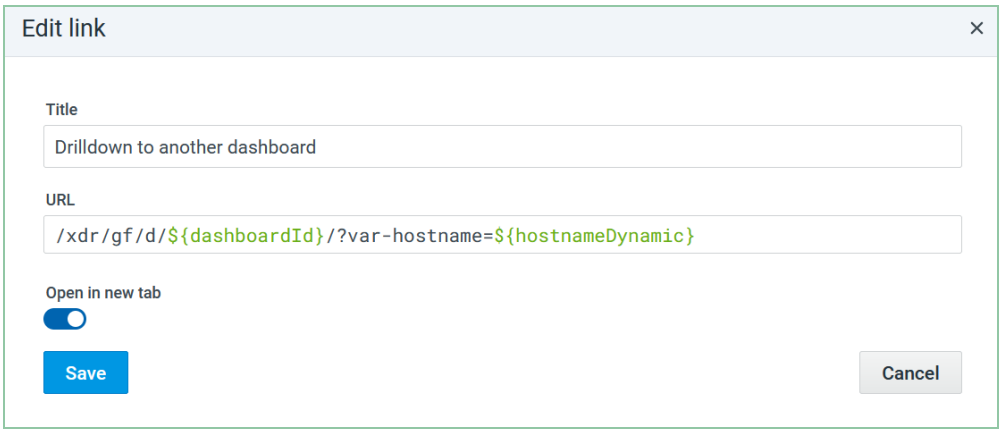
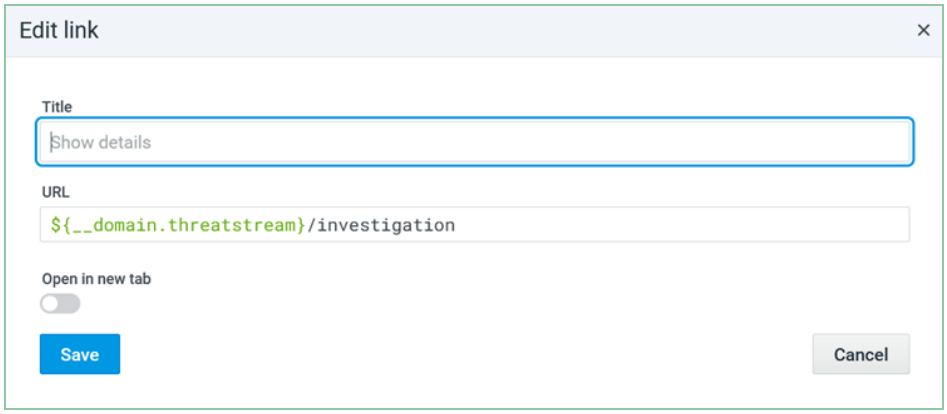
Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Standard options	Unit	Select a unit. Applies to all fields except time.
	Min	Specify a minimum value (as percentile). Leave empty to calculate based on all values.
	Max	Specify a maximum value (as percentile). Leave empty to calculate based on all values.
	Decimals	Number of decimal places in numeric values.
	Displayname	Change the field or series name.
	Novalue	What to show when there is no value.
	Colorscheme	Specify a single color or use the color palette.

Setting		Description
Thresholds	Add threshold	Configure mapping of thresholds to colors.
	Thresholds mode	• Absolute • Percentage
Value mappings	Add value mapping	Click **+Add value mapping** to map a single value or range of values to a specified text string. For example, map data having the country code `SG` to the text `Singapore`.

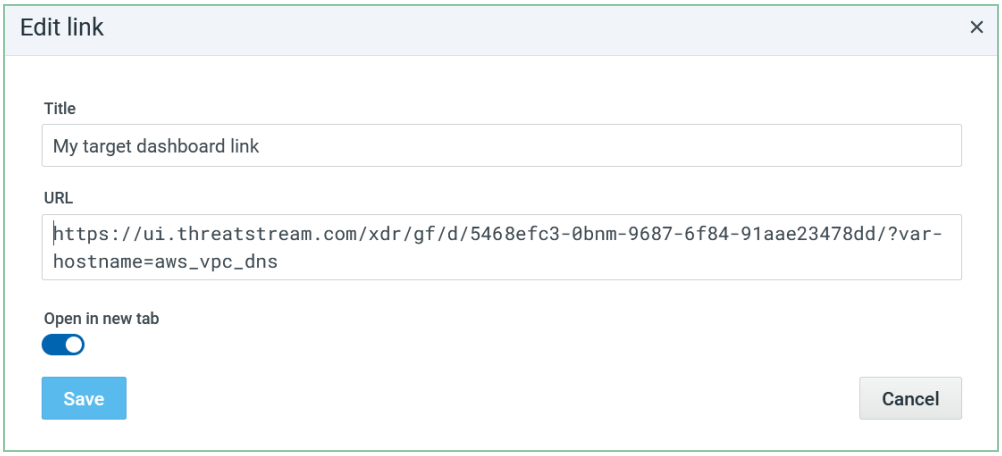
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

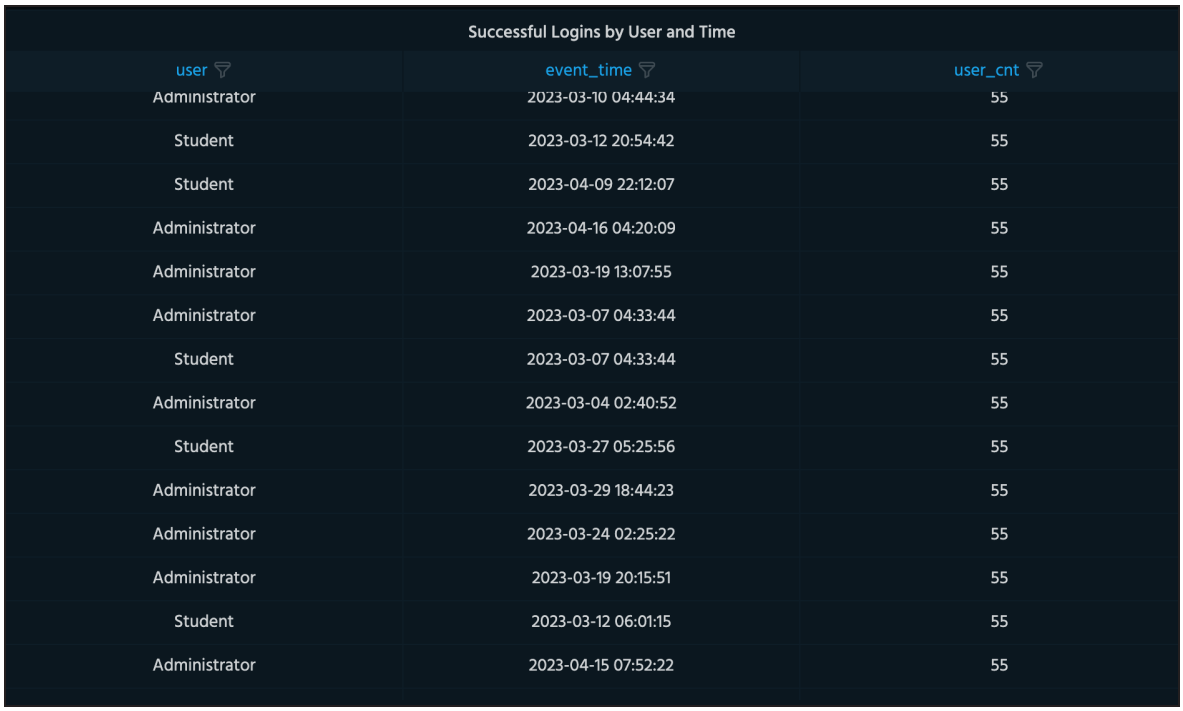
Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

About the Table Visualization Type

Use the Table visualization type to display the columns of the results table in table format.



Successful Logins by User and Time

user	event_time	user_cnt
Administrator	2023-03-10 04:44:34	55
Student	2023-03-12 20:54:42	55
Student	2023-04-09 22:12:07	55
Administrator	2023-04-16 04:20:09	55
Administrator	2023-03-19 13:07:55	55
Administrator	2023-03-07 04:33:44	55
Student	2023-03-07 04:33:44	55
Administrator	2023-03-04 02:40:52	55
Student	2023-03-27 05:25:56	55
Administrator	2023-03-29 18:44:23	55
Administrator	2023-03-24 02:25:22	55
Administrator	2023-03-19 20:15:51	55
Student	2023-03-12 06:01:15	55
Administrator	2023-04-15 07:52:22	55

AQL Expression Requirements

Use the fields operator to specify the table columns.

> **Note:** Using `addtotals=true` in your timechart operator query appends a total row and total column to your results. The `total` row shows the aggregate count per group across the entire time range, and the `total` column shows the aggregate count across all groups per time bucket. This is useful for comparing entities and spotting volume trends, without needing a separate query. See timechart operator for more information.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
Display	Show header	Show table header or not.
Table Editing	Editable toggle	Toggle to allow editing the table.

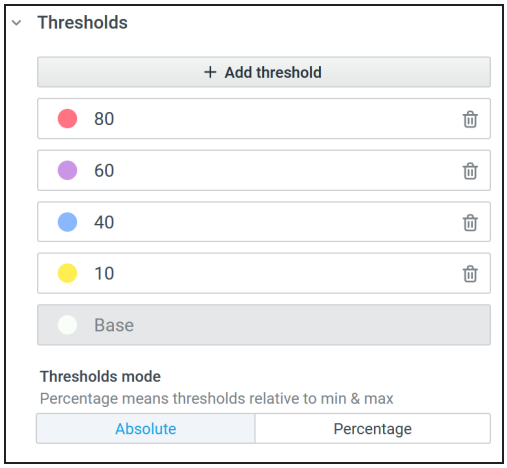
Setting		Description
Links	Add data link	Enter the following data link details:- • Data Link Name: Enter the name of the data link. • Select Column to Link: Select the column you would like to link to the target dashboard, after you click the hyperlinked column entry. • Linked URL: Enter the URL of the target dashboard. This URL contains all the details that will be shared with the target dashboard. **Note:** By default, dashboard timestamps are not included when drilling down. To enable this, you need to configure the drilldown URL by adding the following prefix to the start of the drilldown link: `from=${__from}&to=${__to}&query=<your_search_query>` Here's an example query with the timestamp prefix from above:-

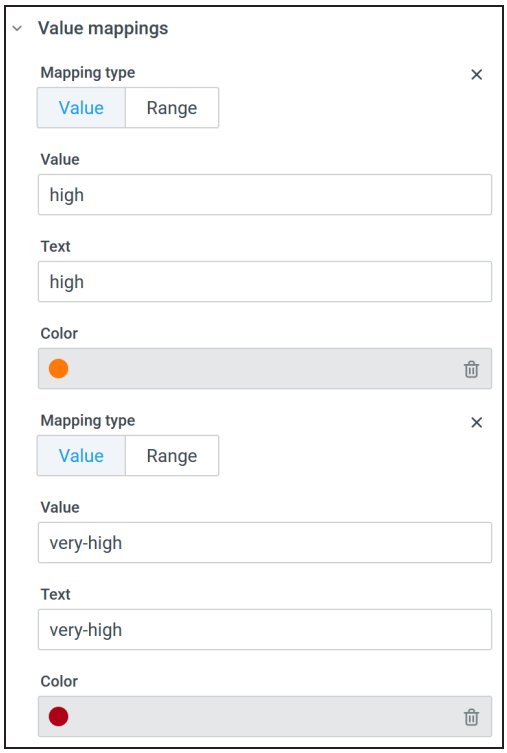
Field Options

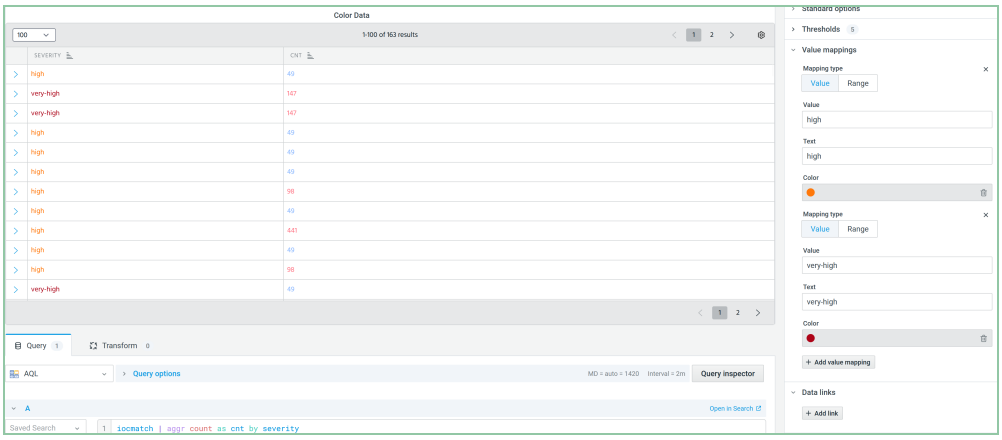
Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Table options	Column width	By default, the width is calculated automatically based on the results data. If desired, specify a fixed width instead.
	Column alignment	• auto • left • center • right
	Cell display mode	Color text, background, show as gauge, etc.
	Column filter	Enables/disables field filters in table.

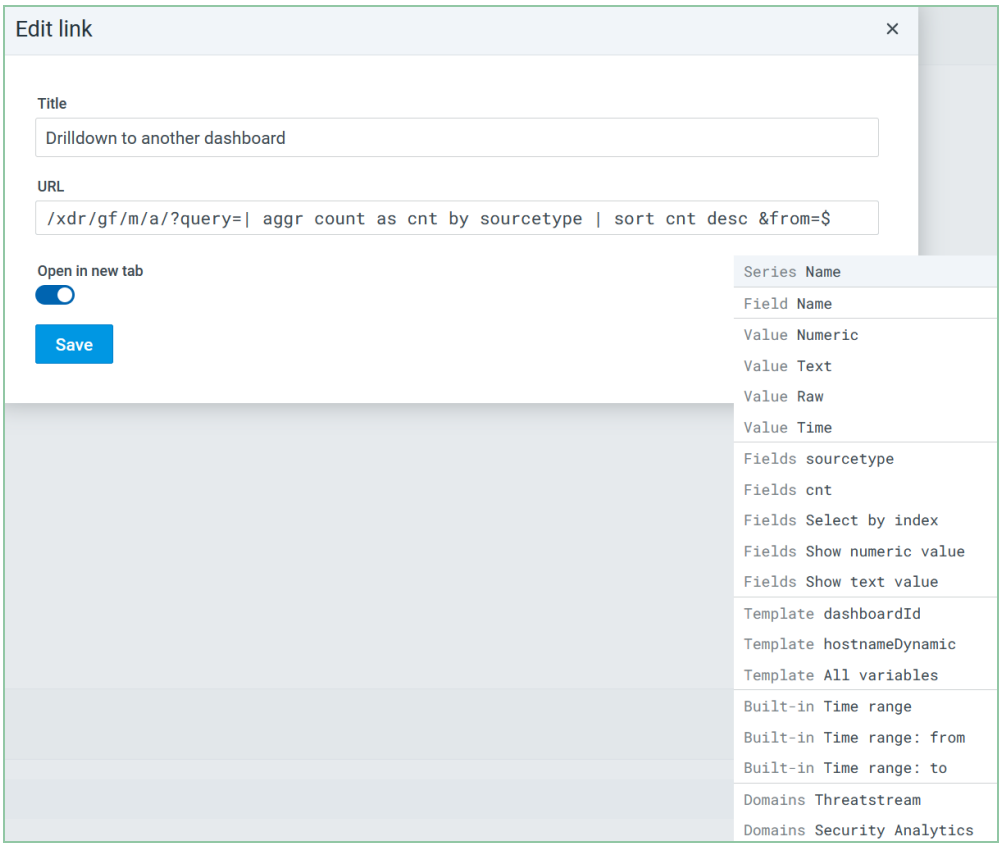
Setting		Description
Standard options	Unit	Select a unit. Applies to all fields except time.
	Min	Specify a minimum value (as percentile). Leave empty to calculate based on all values.
	Max	Specify a maximum value (as percentile). Leave empty to calculate based on all values.
	Decimals	Number of decimal places in numeric values.
	Displayname	Change the field or series name.
	Novalue	What to show when there is no value.
	Colorscheme	Specify a single color or use the color palette.

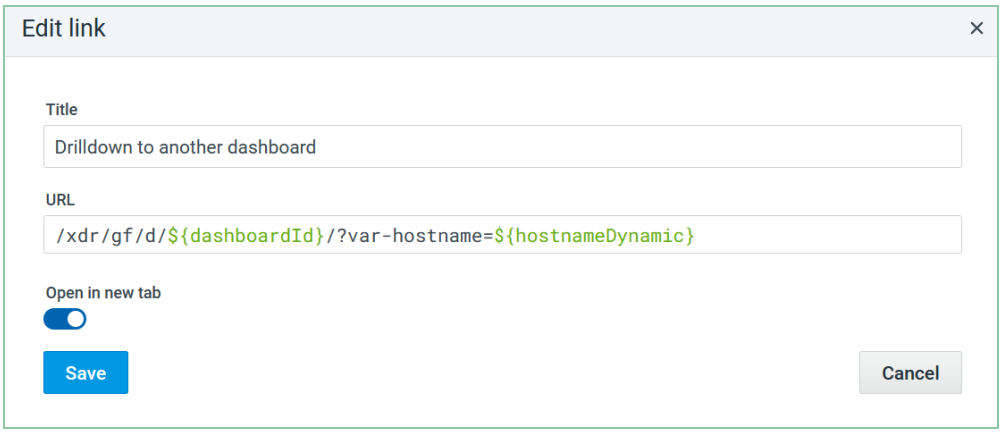
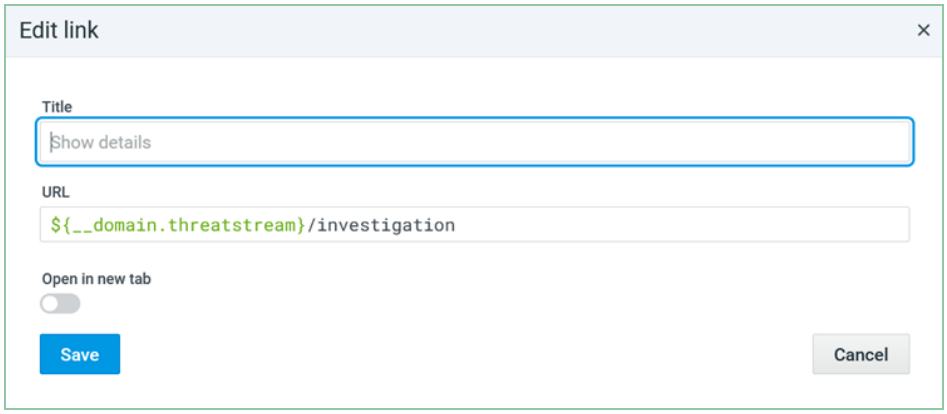
Setting		Description
Thresholds	Add threshold	Configure mapping of thresholds to colors. For example, you can assign blue to threshold values between 40-60 and red to values 80 and higher. Click on the color dot to select different gradients of the same color. 
	Thresholds mode	• Absolute • Percentage

Setting		Description
Value mappings	Add value mapping	Click **+Add value mapping** to map a single value or range of values to a specified text string. For example, map data having the country code `SG` to the text `Singapore`. Additionally, for example, you can color-code your search results depending on the value and colors you assign to values. In the screenshot below, in the Value Mappings section, the severity of an iocmatch is color-coded to indicate high severity results in orange and very-high severity results in red.  These color-coded value mappings, along with the color-coded thresholds (see the Thresholds section) return the following results:

Setting		Description
		

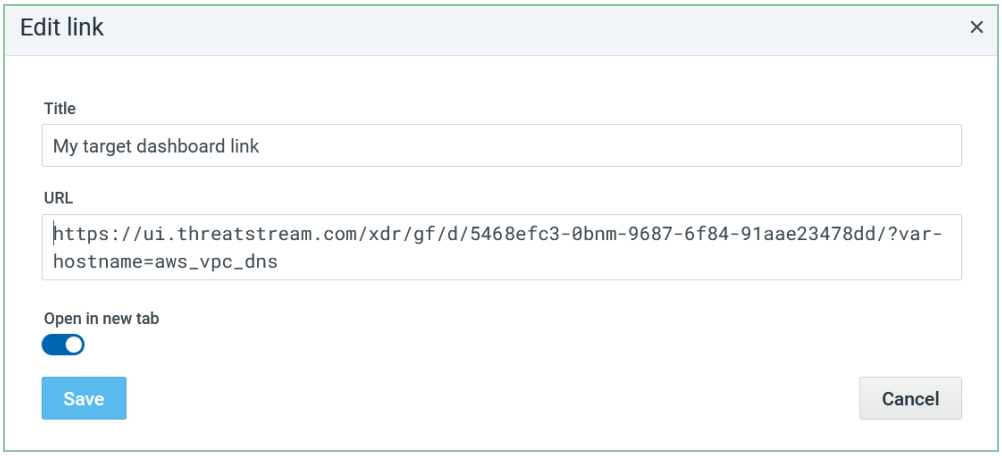
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

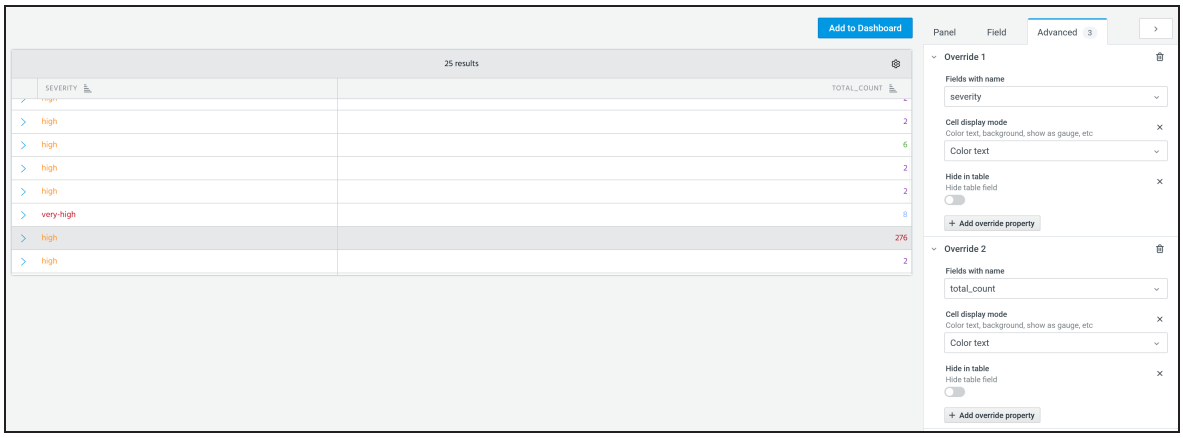
- **Fields with name**: Set properties for a specific field. For example, if you want to color-code different column values as per thresholds, you must add an override property. See "Add override property" below for more information.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

Add override property

To add an override property:

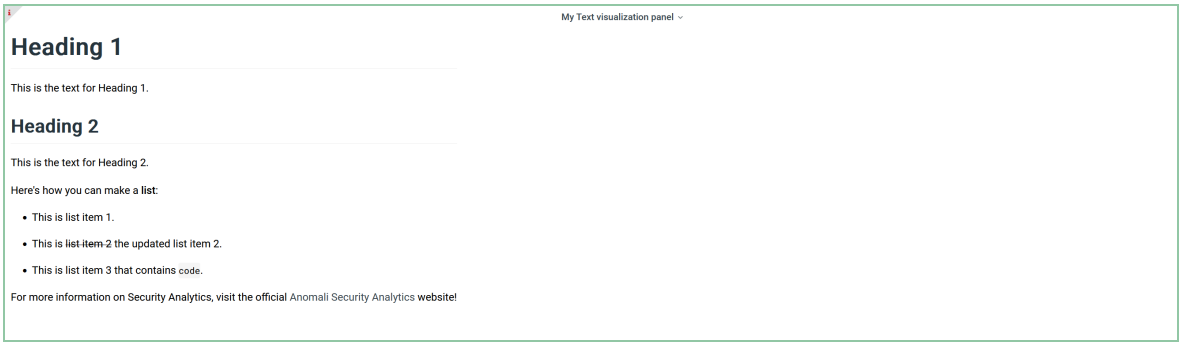
1. Go to the **Advanced** tab.
2. Click **Add an override for**.
3. Select Fields with name and choose a column whose settings you would like to override.
4. Click Add override property and select a property. For example, to color-code different column values as per thresholds, select **Cell display mode** > **Color text**.

Once you configure these settings, your table will have color-coded threshold-based table entries, as follows:



About the Text Visualization Type

Use the Text visualization type to add a dashboard panel that accepts Markdown-formatted input text.



Use case

You can use the Text visualization to:

- Add instructions, descriptions, or reports about using different panels in the dashboard, leading to detailed and well-documented dashboards.
- Capture important contextual information or warning messages you must know to accurately use the dashboard, its workflows, or drilldown capabilities, to name a few.
- Share confidential banners that carry critical information intended for only certain groups of users, helping in dashboard user management.

- Embed images, code snippets, and hyperlinks to create highly informative dashboards that can support documentation and training purposes.

Panel Options

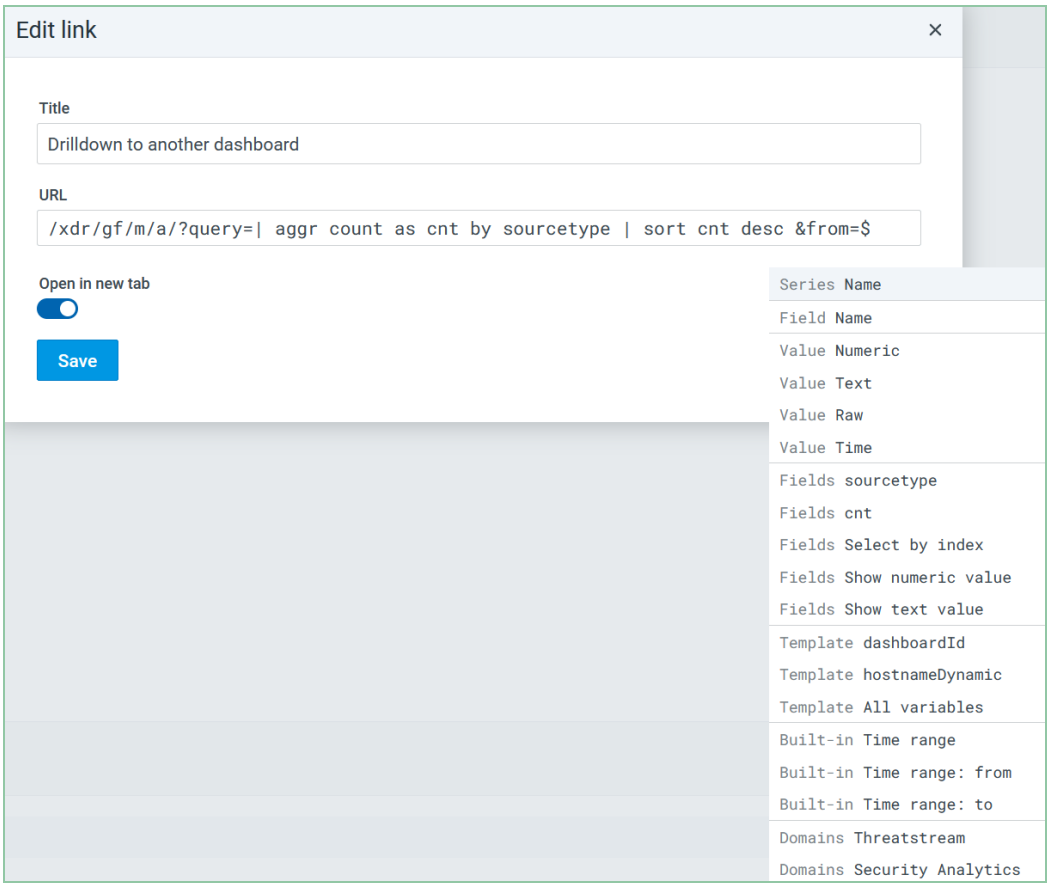
Use the options in the Panel tab to customize the options described in the following table.

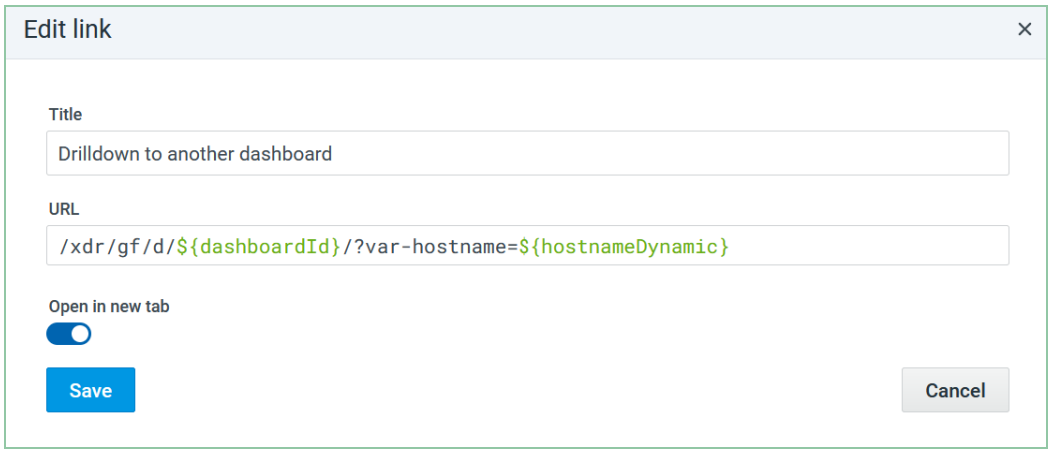
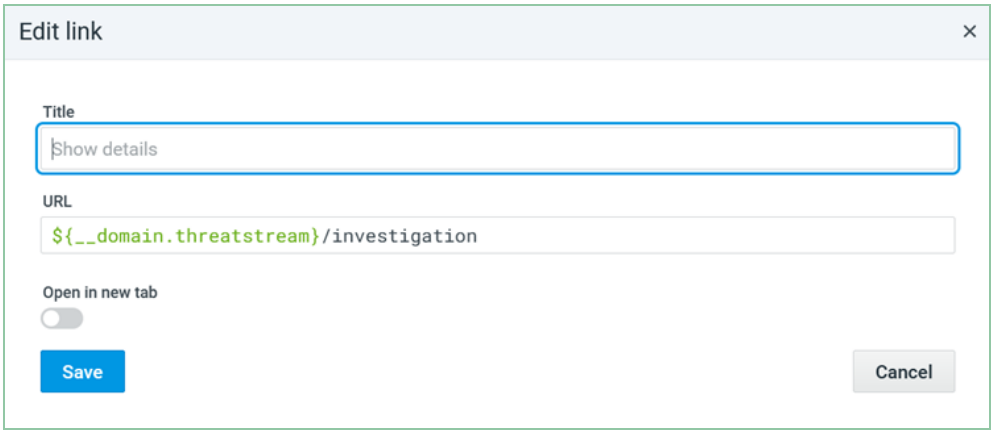
Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.

Field Options

Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

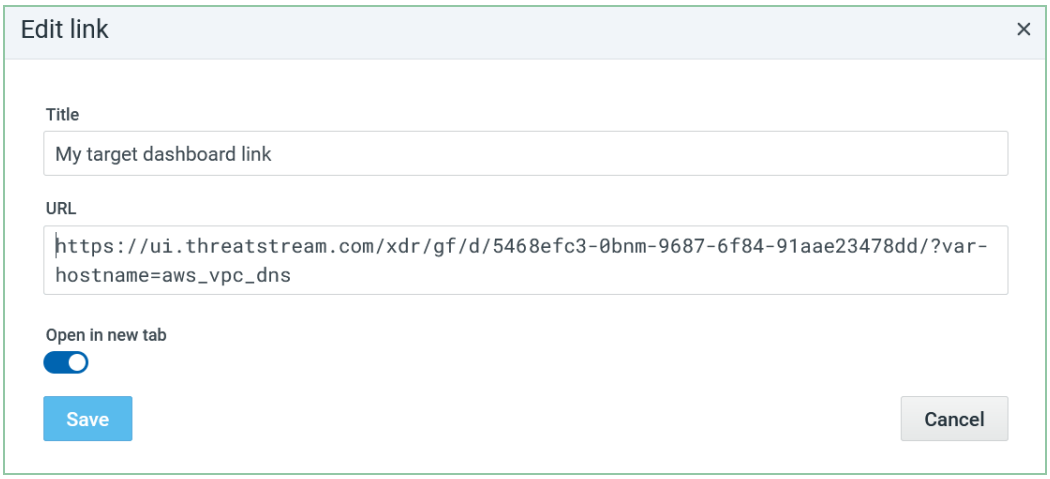
Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign ($) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("/"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**. 

Setting		Description
		In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream. **Add link to another dashboard** To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>`

Setting		Description
		Multiple variables: Multiple context variables and their corresponding values, separated by ampersand (&).

```
<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>
```



In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.

Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:

Setting		Description
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

Once you add this visualization type to your dashboard, you can also manage your dashboard panels.

See Manage dashboard panels to learn more about how you can edit panel settings to modify the visualization and apply advanced transformations, share panels with permitted users in your organization, or even view the panel JSON model, and so on.

Support for Variables

The Markdown widget also supports dynamic variables on Anomali Match Dashboards. You can now insert both system-defined and custom variables into Markdown content, enabling the display of dynamic, context-sensitive information

(for example, usernames, dates, alert counts) that updates automatically based on AQL data or user-defined inputs.

How to use variables

Variables can be inserted into your Markdown content using double curly braces:

```
${enter_your_variable_name}
```

When the widget is rendered, these placeholders are replaced with their corresponding values.

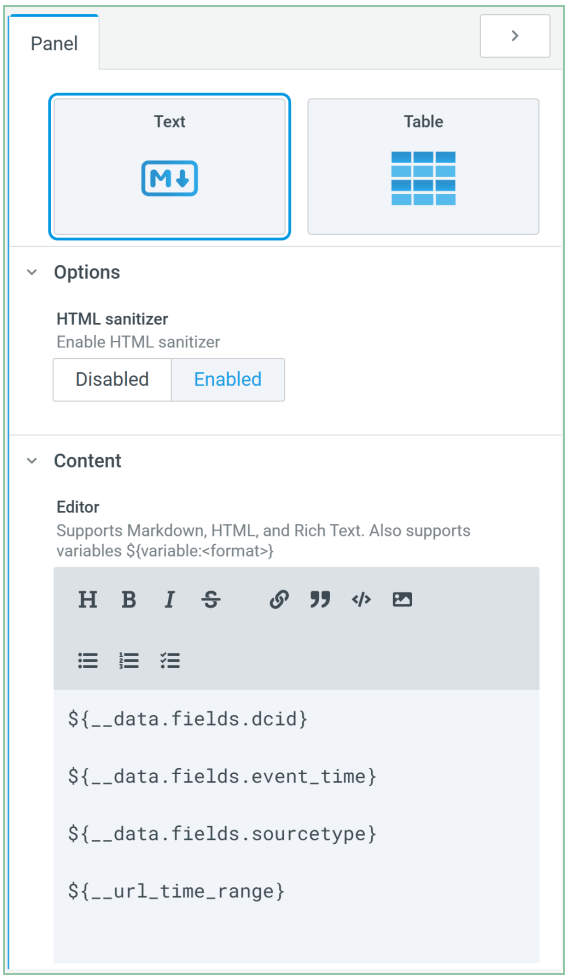
Adding a variable

See Adding Dashboard Variables to understand how you can add and configure variables.

Dynamic Usage of Variables

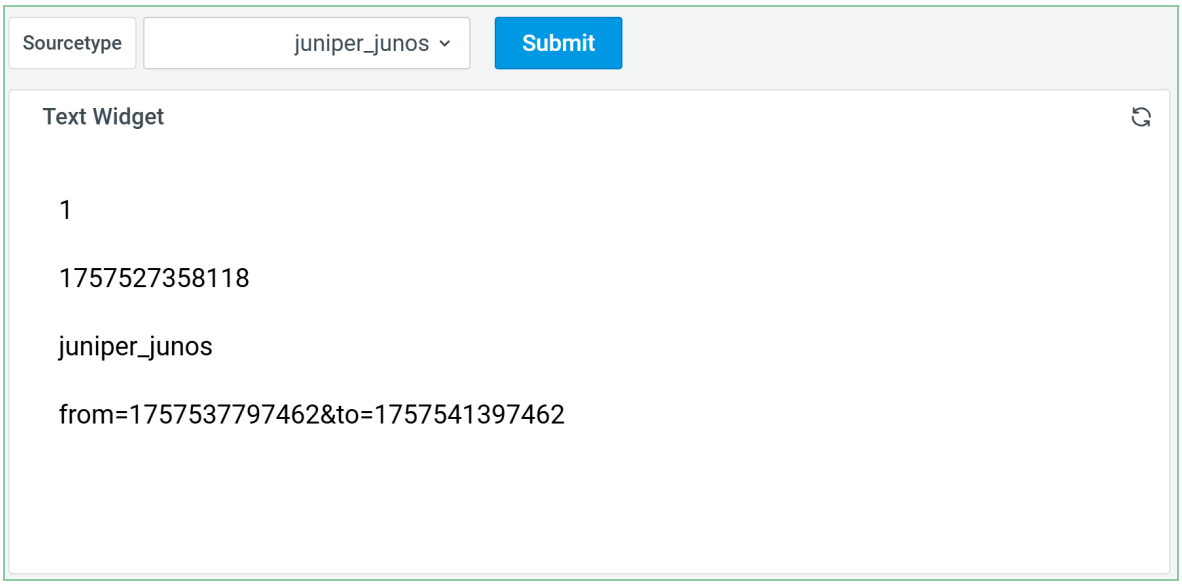
After "Adding a variable" above on a dashboard, you can dynamically display their context-specific values as follows:

1. Click **Edit** on the dashboard.
2. Hover your mosue to the right of the Panel title and click the expander. Doing so displays the panel management menu.
3. Click **Edit**.
4. In the Content Editor, enter the dollar sign ($). This sign displays all the system and custom variables that you can use to dynamically display context-specific information. In the example below, we add variables for dcid, event time, sourcetype, and time range.



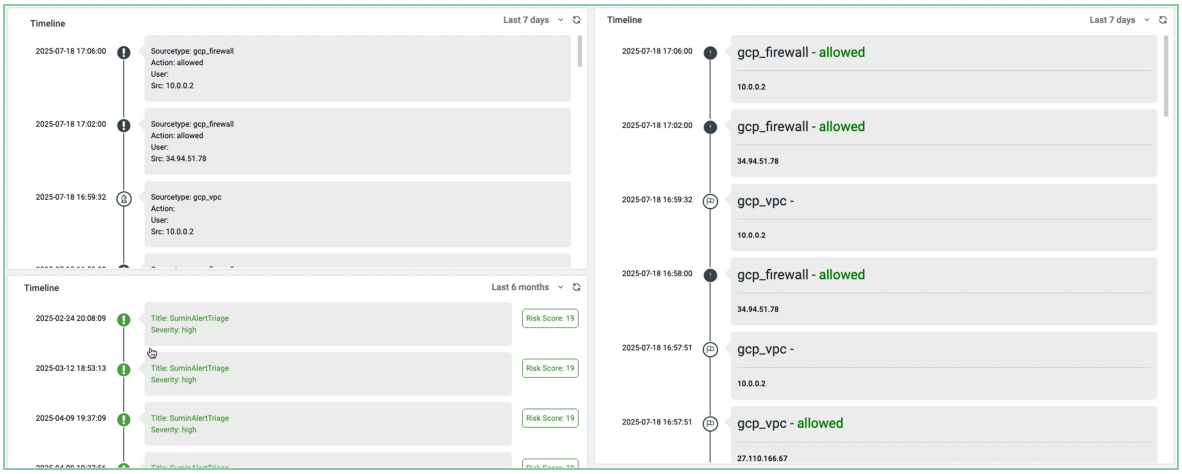
5. Click **Save**.

Once you complete these steps, the variable you added will be dynamically populated with its value(s), based on AQL data or user-defined inputs.



About the Timeline Visualization Type

Use the Timeline visualization type to visualize user and entity behavior in a timeline view, enabling you to quickly identify, investigate, or triage high-risk users and assets through an intuitive timeline interface.



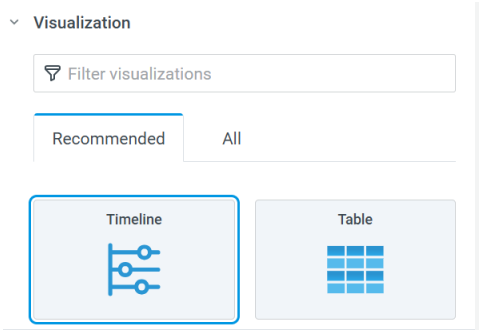
Use case

With the Timeline visualization, you can drill-down to detailed views, such as:

- **Asset timeline**: With further drill-down to search, this timeline can capture assets most at risk based on data from IOC matches and triggered alerts.
- **User timeline**: With further drill-down to search, this timeline can highlight users with the highest risk scores, based on both IOC matches and alert activity.
- **Alert timeline (Top alerts triggered)**: With drill-down to alert triage and then to search, this timeline lets you pivot from high-level summaries to specific alert details and related search queries.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
	Show Description	Display or hide the description.
Visualization	Timeline	Select the Timeline widget to create timelines. 

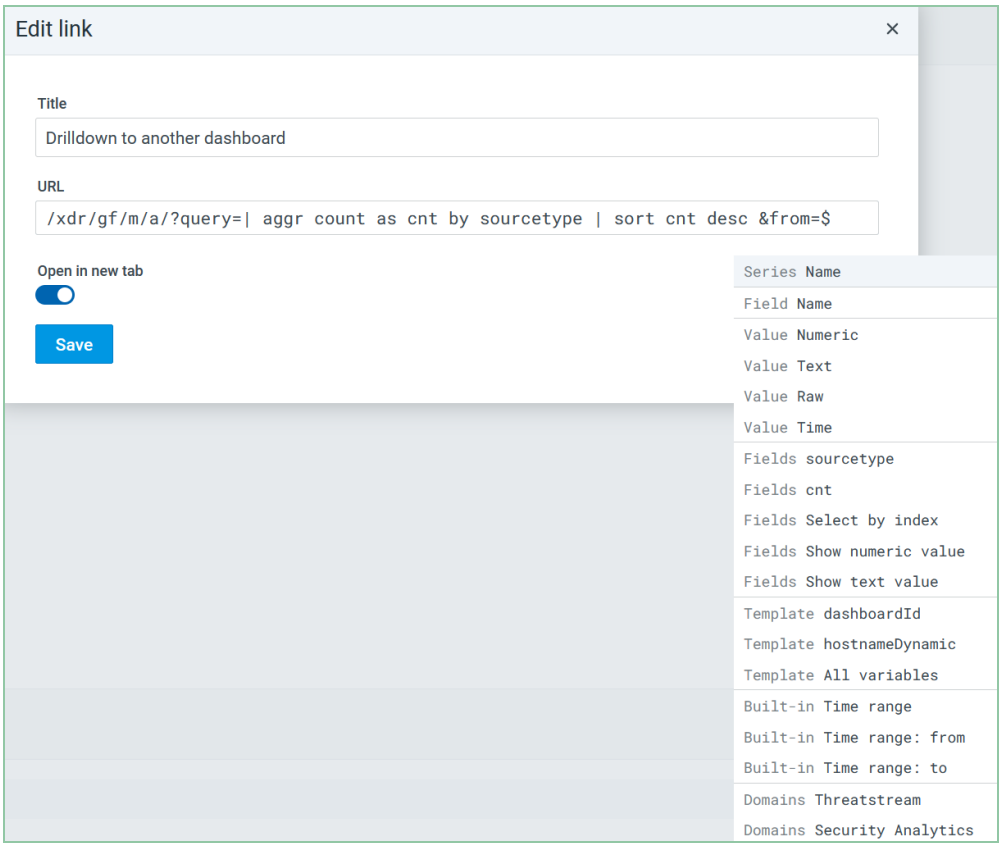
Setting		Description
Fields	Date and Time	Map date to any date and time field in the search result.
	Field for Icon Mapping	Select a field whose values are mapped to a predefined list of icons. Click **Add Icon Map** to add an icon map.
	Message	Add a timeline message that supports variables. Use the dollar-sign ($) to invoke system variables that change dynamically based on AQL data or user-defined inputs. See "Support for Variables" on page 381 for more information.
	Score	Map score to any numeric field within the search result.

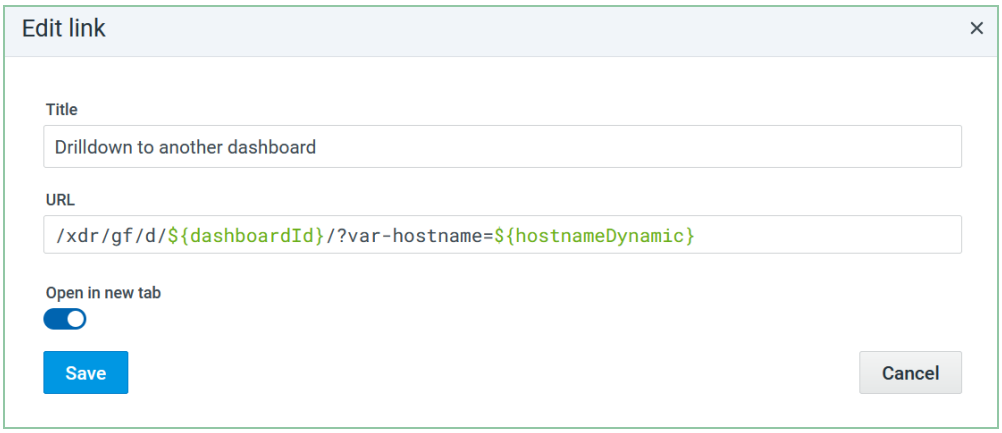
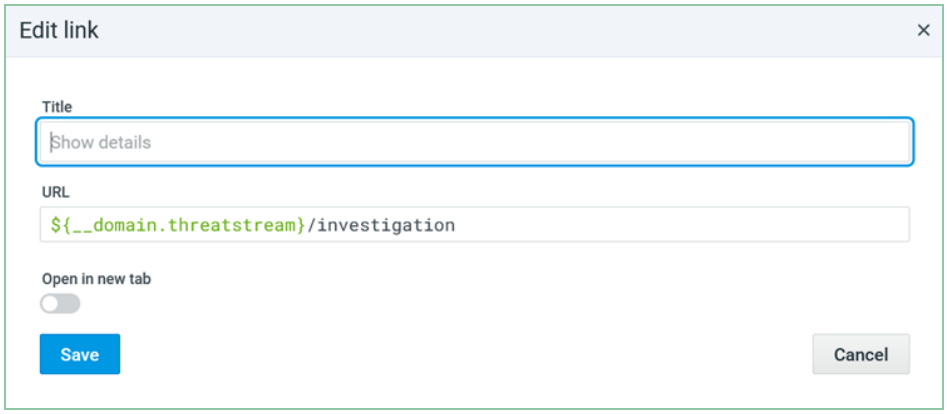
Field Options

Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Thresholds	Add threshold	Configure mapping of thresholds to colors.
	Thresholds mode	• Absolute • Percentage

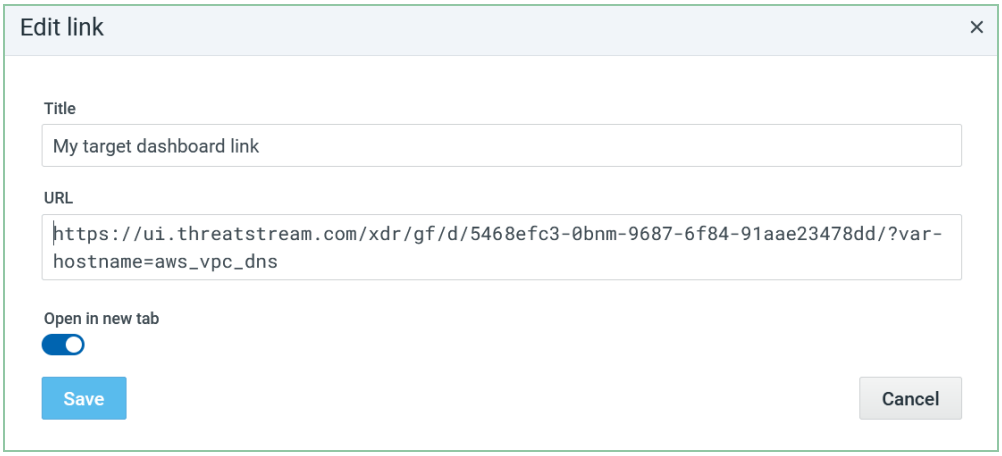
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

Once you add this visualization type to your dashboard, you can also manage your dashboard panels.

See Manage dashboard panels to learn more about how you can edit panel settings to modify the visualization and apply advanced transformations, share panels with permitted users in your organization, or even view the panel JSON model, and so on.

Support for Variables

The Timeline widget also supports dynamic variables on Anomali Match Dashboards. You can define and use both system-defined and custom variables in the **Message** text field, enabling the display of dynamic, context-sensitive information (for example, usernames, dates, alert counts) that is automatically updated based on AQL data or user-defined inputs.

How to use variables

Variables can be inserted as a message using double curly braces:

```
${enter_your_variable_name}
```

When the widget is rendered, these placeholders are replaced with their corresponding values.

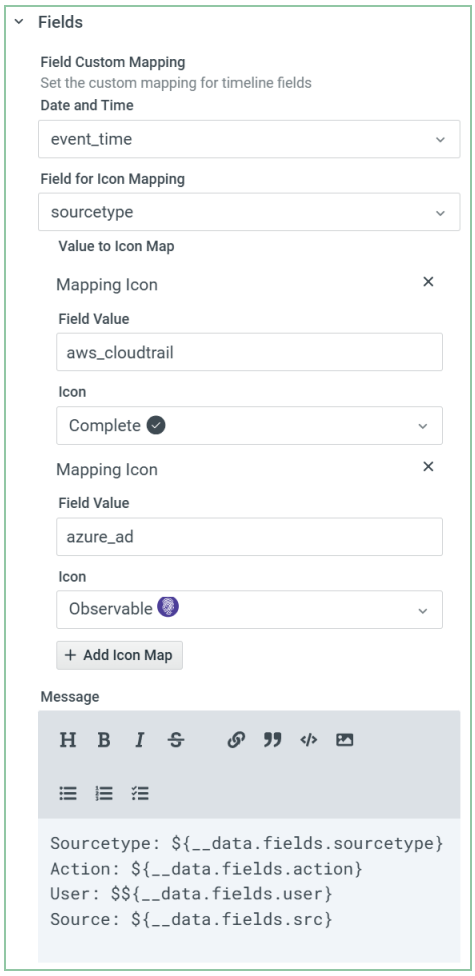
Adding a variable

See Adding Dashboard Variables to understand how you can add and configure variables.

Dynamic Usage of Variables

After "Adding a variable" on the previous page on a dashboard, you can dynamically display their context-specific values as follows:

1. Click **Edit** on the dashboard.
2. Hover your mouse to the right of the Panel title and click the expander. Doing so displays the panel management menu.
3. Click **Edit**.
4. In the **Message** editor, enter the dollar-sign ($). This sign displays all the system and custom variables that you can use to dynamically display context-specific information. In the example below, we add variables for sourcetype, action, user, and source IP address.



5. Click **Save**.

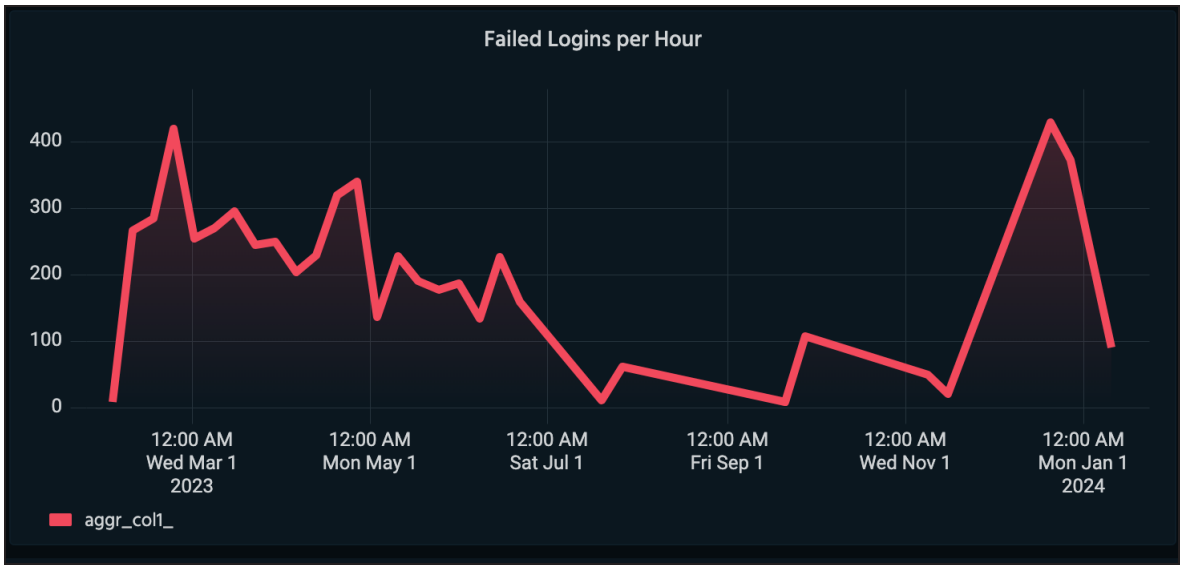
Once you complete these steps, the variable you added will be dynamically populated with its value(s), based on AQL data or user-defined inputs.

Note: For fast and optimized performance, Anomali recommends that the timeline widget be used with up to 50 rows results with 5

columns.

About the TimeSeries Visualization Type

Use the TimeSeries visualization type to show trends over time.



AQL Expression Requirements

The timeseries visualization type is used to show counts over time. The query must have the following elements to return the required data:

- A result set having an event_time column and a numeric column.
- Use the `timetrend` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

```
| timetrend count as cnt span=1h | sort event_time desc
```

Note: Using `addtotals=true` in your timechart operator query appends a total row and total column to your results. The `total` row shows the aggregate count per group across the entire time

> range, and the `total` column shows the aggregate count across all groups per time bucket. This is useful for comparing entities and spotting volume trends, without needing a separate query. See timechart operator for more information.

Panel Options

Use the options in the Panel tab to customize the options described in the following table.

Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
Display	Tooltip mode	• Single • All • Hidden
	Legend mode	• List • Table • Hidden
	Legend placement	• Bottom • Right
	Legend calculations	Select functions. Advanced users only.

Field Options

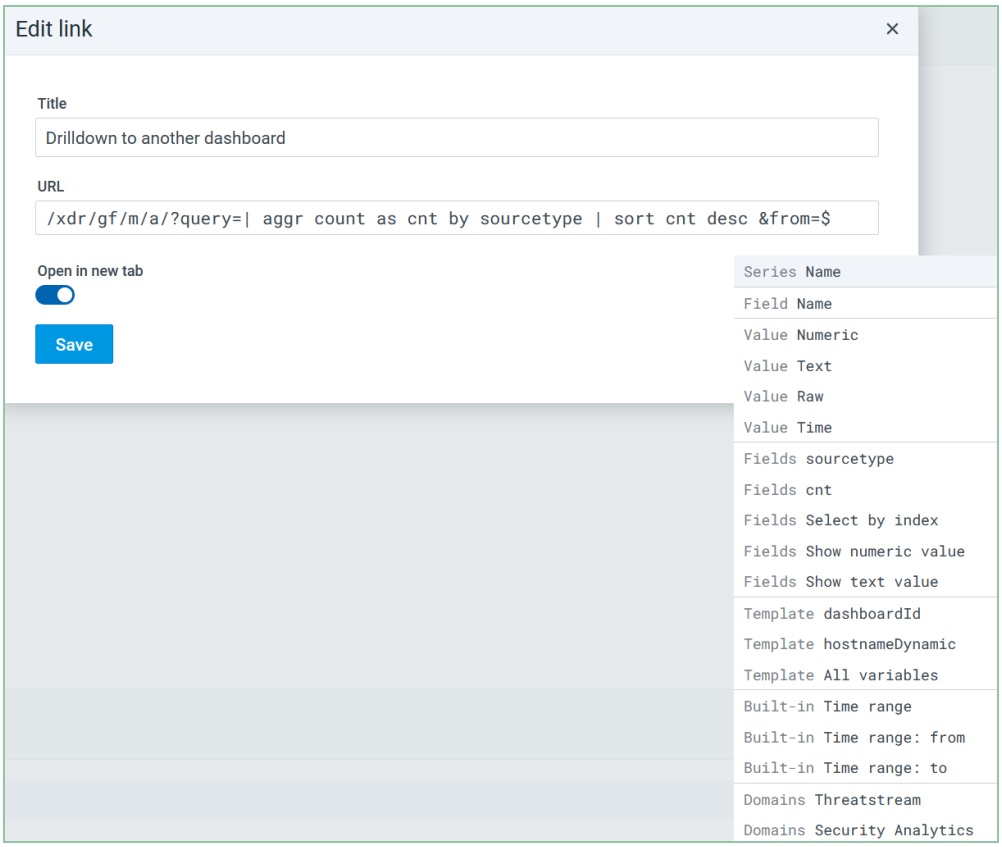
Use the options in the Fields tab to customize the options described in the following table.

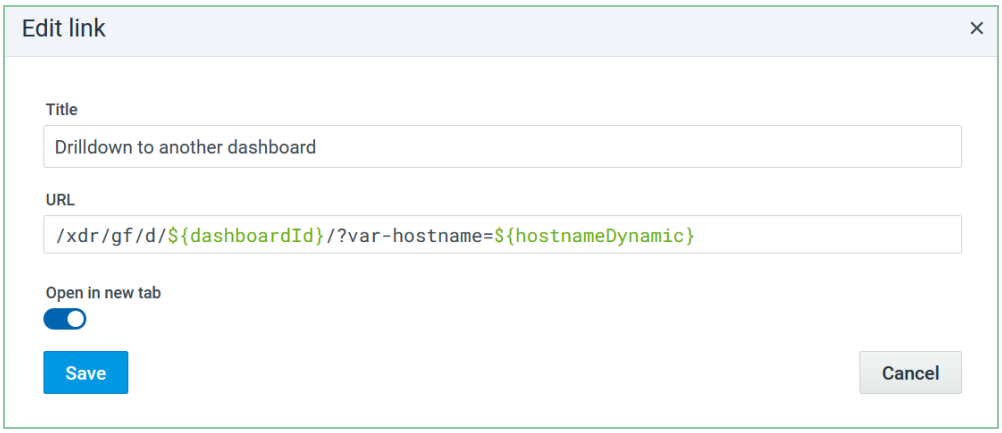
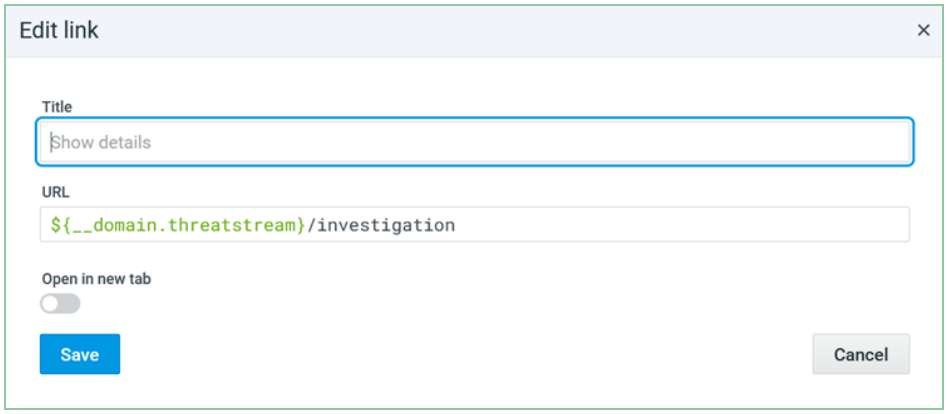
Setting		Description
Time Series options	Style	• Lines • Bars • Points Additional style options are displayed after you select a style. The preview is updated instantly to show the impact from changing style options.
Axis	Placement	• Auto • Left • Right • Hidden
	Label	Additional label for y-axis.
	Width	By default, the width is calculated automatically based on the results data. If desired, specify a fixed width instead.
	Soft min Soft max	By default, the Y-axis limits are determined automatically based on the results data. Adjust these settings if the automatic behavior distorts the visualization.
	Scale	• Linear • Logarithmic

Setting		Description
Standard options	Unit	Select a unit. Applies to all fields except time.
	Min	Specify a minimum value (as percentile). Leave empty to calculate based on all values.
	Max	Specify a maximum value (as percentile). Leave empty to calculate based on all values.
	Decimals	Number of decimal places in numeric values.
	Display name	Change the field or series name.
	No value	What to show when there is no value.
	Color scheme	Specify a single color or use the color palette.

Setting		Description
Thresholds	Add threshold	Configure mapping of thresholds to colors.
	Thresholds mode	• Absolute • Percentage
Value mappings	Add value mapping	Click **+Add value mapping** to map a single value or range of values to a specified text string. For example, map data having the country code `SG` to the text `Singapore`.

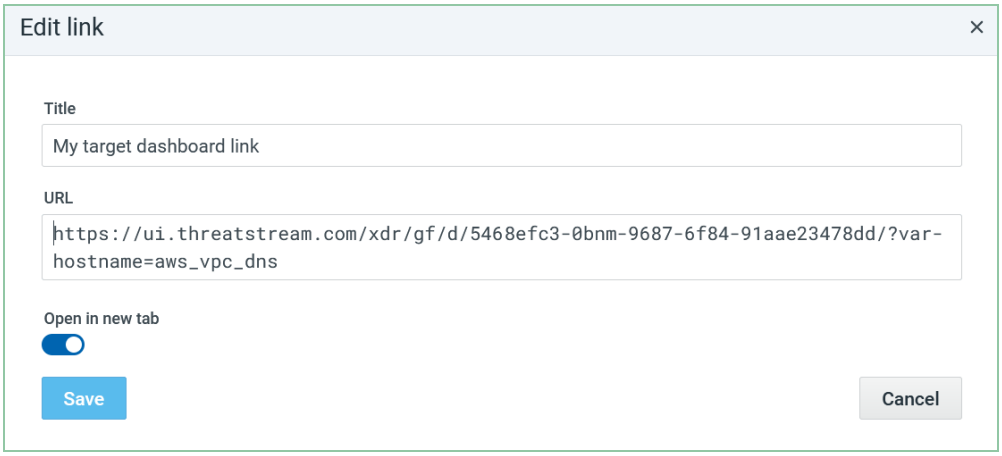
Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign (`$`) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("`/`"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**.  In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream.

Setting		Description
		Add link to another dashboard To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>` **Multiple variables**: Multiple context variables and their corresponding values, separated by ampersand (`&`).

Setting		Description

		`<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`
		
		 
		
		In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`.
		
		Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:
		
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

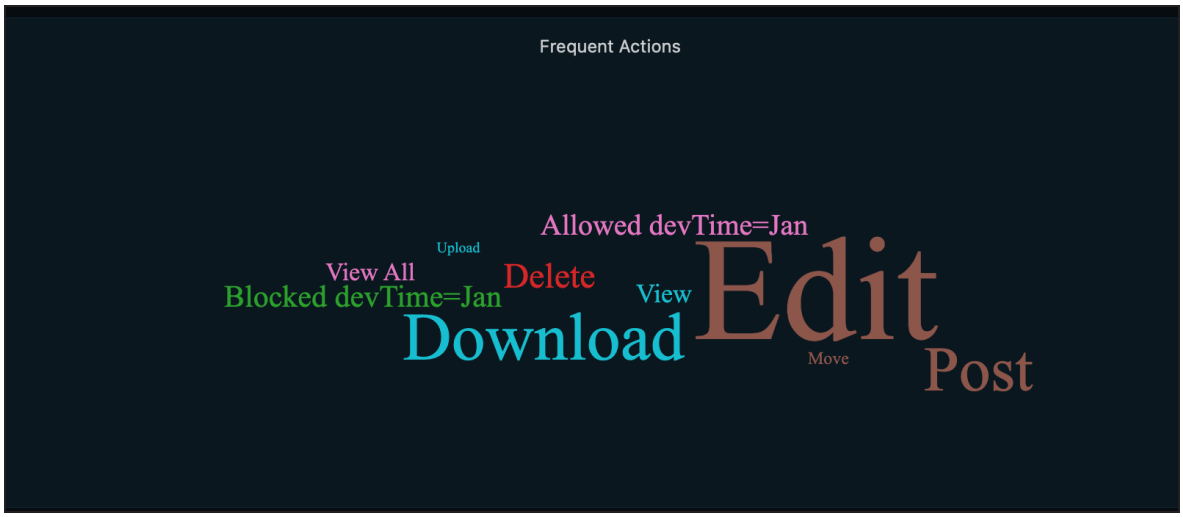
Use the options in the Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

About the Wordcloud Visualization Type

Use the Wordcloud visualization type to show the prevalence of specific values for a field.



AQL Expression Requirements

The wordcloud visualization type is used to reflect relative counts. The query must have the following elements to return the required data:

- A result set having a numeric column.
- Use the `aggr` operator, which returns a numeric value.

The following is an example of a query that returns the required data:

```
|aggr count as act_cnt by action
|sort act_cnt desc
|limit 10
```

Panel Options

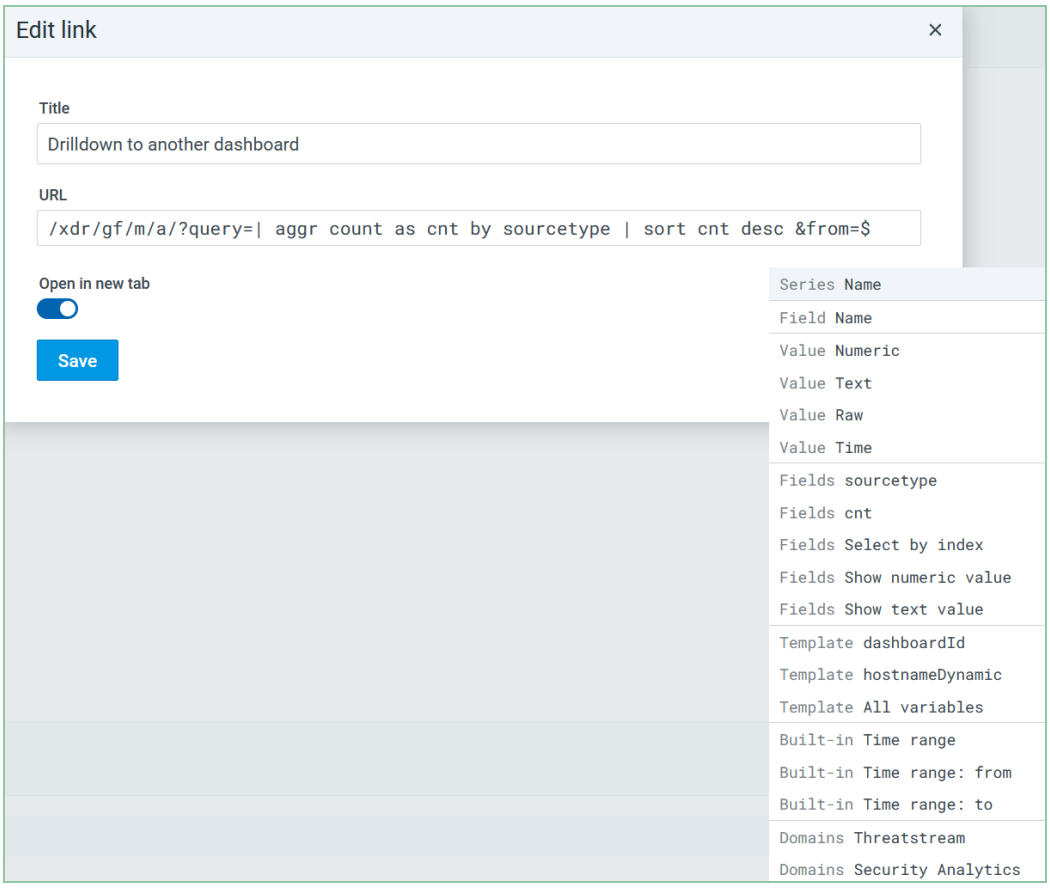
Use the options in the Panel tab to customize the options described in the following table.

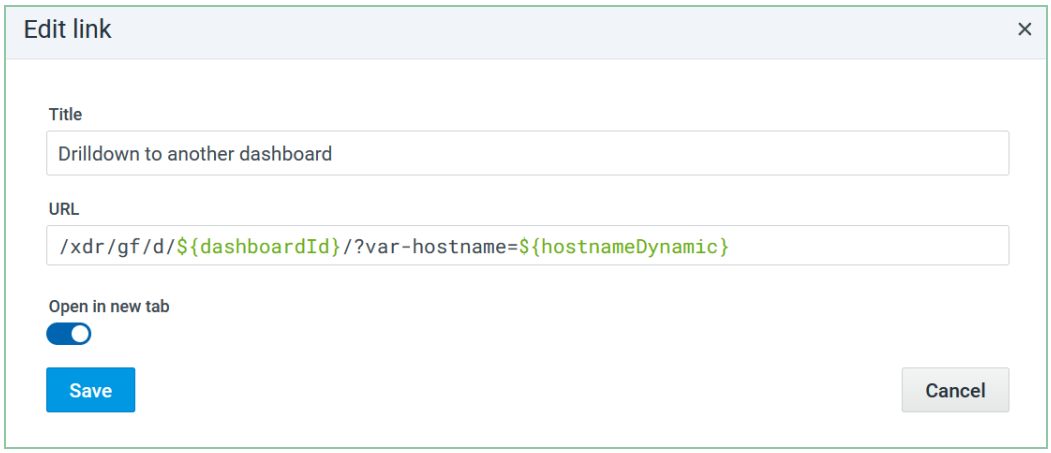
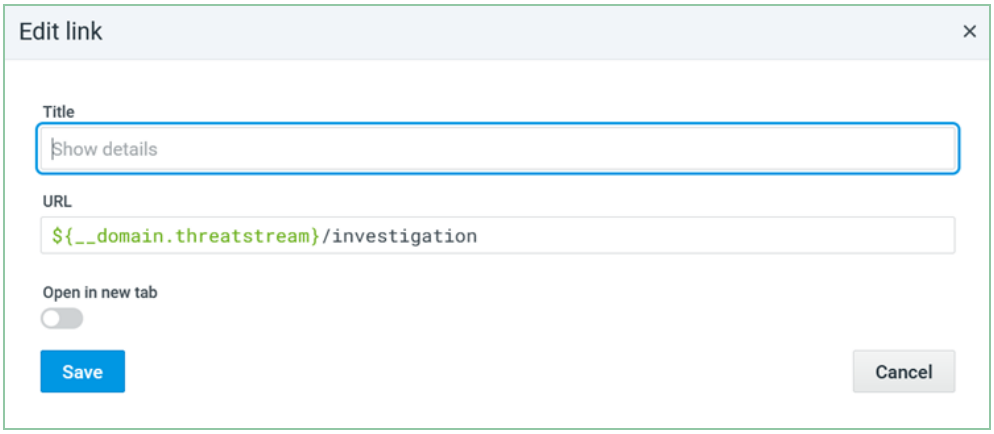
Setting		Description
Settings	Panel title	Titles appear at the top of a panel.
	Description	Descriptions appear as tooltips. Markdown and links are supported.
	Transparent	Display panel without a background.
Display	Words Field	Query field having values to be rendered.
	Count Field	Aggregation field.
	Padding	Padding between words.
	Font min size	Font size for smallest count.
	Font max size	Font size for largest count.
	Min rotation angle Max rotation angle	Alter the rotation angle of the words.
	Rotation steps	Number of rotation steps.
	Deterministic	By default, words are positioned and rotated randomly. Select this option to have similar inputs treated the same.

Field Options

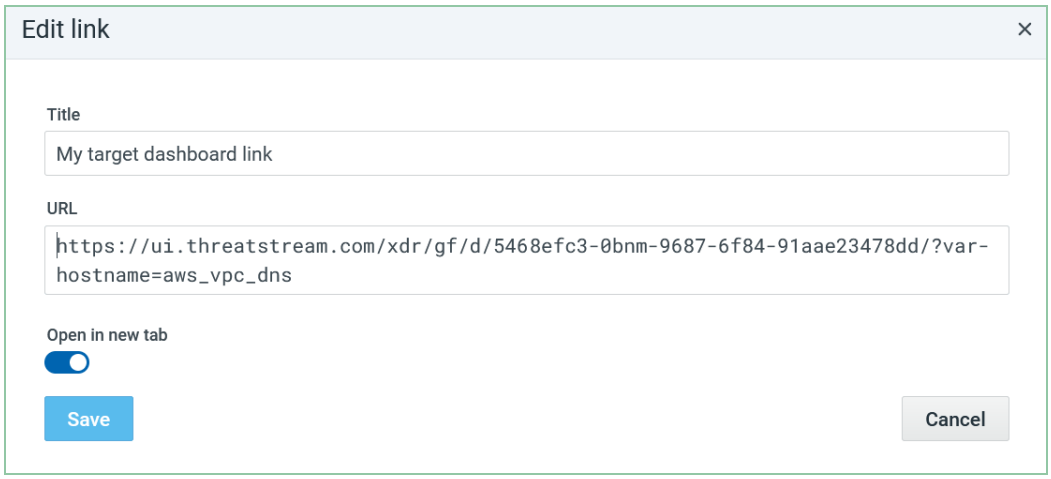
Use the options in the Fields tab to customize the options described in the following table.

Setting		Description
Data links	Add link	Click **+Add link** and complete settings for link behavior when a user clicks data fields in the visualization. You can use the following URL types and configure them to pass variables: • Relative URL – URL that will open a new Search page with a specified search query. • URL in ThreatStream or Security Analytics domains – URL that will open a page or a specific object in ThreatStream or Security Analytics. You can use a relative URL in either of the following ways: • Specify `/xdr/gf/m/a/?query=` in the URL field, and then provide a search query. Then, type a dollar-sign (`$`) to display a list of variables that you can insert into the query.

Setting		Description
		 • Specify `/xdr/gf/d/` in the URL field, then directly type a dollar-sign (`$`) to use a reference to different variables in the dashboard.

Setting		Description
		 To use a URL in ThreatStream or Security Analytics domains, do the following: 1. Type a dollar-sign ($) in the URL field to display a selection list of variables that can be inserted into the URL pattern, and then select `Domains Threatstream` or `Domains Security Analytics`. 2. Using a forward slash ("/"), specify the page or a link to an object in ThreatStream or Security Analytics, and then click **Save**. 

Setting		Description
		In the example above, clicking on any part of the visualization opens the Investigations page in ThreatStream. **Add link to another dashboard** To use the data in a dashboard panel as context for navigating to a different dashboard and populating its panels in ThreatStream or Security Analytics: 1. Go to the **Dashboard** library, open your target dashboard, and click the **Share dashboard or panel** icon ( ) on the top-left. 2. Copy the **Link URL** displayed on screen. This is the URL for the target dashboard. 3. On your current dashboard, in the **Edit link** form, paste the **Link URL** you obtained in Step 2. 4. After the target dashboard ID, enter the `/?` symbols, then the string `var-`, followed by the dashboard variable and its value that you would like to use as context to link your current dashboard to the target dashboard and populate its panels. 5. Enter a title for the target dashboard. Optionally, enable the **Open in new tab** switch to ensure the target dashboard opens in a new tab. 6. Click **Save**. When done correctly, the data link would have the following formats: **Single variable**: Single context variable with its value. `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable>=<my-context-variable-value>`

Setting		Description
		Multiple variables: Multiple context variables and their corresponding values, separated by ampersand (&). `<domain_you_selected>/xdr/gf/d/id-of-the-target-dashboard/?var-<my-context-variable1>=<my-context-variable1-value>&var-<my-context-variable2>=<my-context-variable2-value>`  In the example in the screenshot above, clicking on a linked dashboard panel will carry the context through `hostname=aws_vpc_dns` to the target dashboard with dashboard ID = `5468efc3-0bnm-9687-6f84-91aae23478dd`, and populate its panels with information related to `aws_vpc_dns`. Alternatively, instead of entering the value in Step 4, you can also type a dollar-sign (`$`) in the URL and use the reference to the field. Using `hostname` from the aforementioned example and referencing its value, the data link becomes:

Setting		Description
		`https://ui.threatstream.com/xdr/gf/d/5468efc3-0bnm-9687-6f84-91aae23478dd/?var-hostname=${__data.fields.hostname}`

Advanced Options

Use the options in Advanced tab to override the settings configured in the Fields tab for particular fields. For example, you can override the automatic color palette selected for a particular field to a preferred single color.

You have the following options to select field settings to be overridden:

- **Fields with name**: Set properties for a specific field.
- **Fields with name matching regex**: Set properties for fields with names matching a regex.
- **Fields with type**: Set properties for fields with a specific type (numeric, string, Boolean).
- **Fields returned by query**: Set properties for fields from a specific query.

Chapter 16: Anomali Query Language Reference

Anomali Query Language (AQL) is a piped language, similar to Splunk and Kusto.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`. See "OCSF Schema Overview" on page 39 for more information.

> **Note:** AQL boolean arguments are case-insensitive, that is, all the following boolean arguments are valid:
>
> **Positive case**: `True`, `true`, `t`
>
> **Negative case**: `False`, `false`, `f`

AQL Cheat Sheet

The following cheat sheet provides reference tables that compare the most frequently used AQL operator queries to their equivalent Kusto or Splunk queries.

> **Note:** All operators shown work with both `eventlog` and `ocsf` tables, unless otherwise noted. For OCSF field syntax, see "OCSF Schema Overview" on page 39.

Filter Search

	Anomali	Kusto	Splunk
Operator	`"where" on page 700`	`where`	`search`

	Anomali	Kusto	Splunk
Example (`eventlog`)	`| where sourcetype contains "CrowdStrike" and image endswith "\pnputil.exe"`	`|where Session_Id == "c8894ffd-e684-43c9-9125-42adc25cd3fc" and ingestion_time() > ago(24h)`	`Event.Rule="330009.2" Session.Id="c8894ffd-e684-43c9-9125-42adc25cd3fc" _indextime>-24h`
Example (`ocsf`)	`ocsf | where class_name = "Network Activity" and src_endpoint.ip = "192.168.1.10"`	—	
Notes	• Anomali search starts with an implicit `select` from the `eventlog` table. • Refer to the Common Event Schema for descriptions of fields and example values.	Kusto log queries start from a tabular result set in which a filter is applied.	In Splunk, filtering is the default operation on the current index.

Select Results Table Columns

	Anomali	Kusto	Splunk
Operator	`"fields" on page 533`	`facets`	`fields`
Example (`eventlog`)	`| fields event_time, src, image, command_line`	`|facet by App_Branch, App_Version`	`|fields App.Version, App.Platform`
Example (`ocsf`)	`ocsf | fields src_endpoint.ip, actor.user.name`	—	

Aggregate Rows

	Anomali	Kusto	Splunk
Operator	"aggr" on page 436	summarize, join	stats, eventstats
Example (eventlog)	| aggr count by src	|summarize count() by App_Platform, Release_Audience	|stats count by OSEnv, Audience
		|join kind=inner (T1) on _time |project _time, category, count_i, count_total	… |stats count_i by time, category |eventstats sum (count_i) AS count_total by _time_
Example (ocsf)	ocsf | aggr count by src_endpoint.ip, dst_endpoint.port	—	

Aggregation over Timespan

	Anomali	Kusto	Splunk
Operator	"timetrend" on page 682	summarize	timechart
Examples	| timetrend count span=1mon | timetrend count by sourcetype span=10m	summarize EventCount=count() by EventType, Length=bin (Duration, 1d)	timechart span=1m avg (CPU) BY host

Limit Results

	Anomali	Kusto	Splunk
Operator	"limit" on page 597	limit	head
Example	| limit 30	|limit 30	|head 30

Sort Results Table

	Anomali	Kusto	Splunk
Operator	"sort" on page 653	order by	sort
Example	| sort count desc	|order by Data_ Hresult, desc	|sort Data.Hresult |reverse

Calculate Fields

	Anomali	Kusto	Splunk
Operator	"calc" on page 461	extend or project	eval
Example	| calc result = case(status = 200, 'OK', status = 404, 'Not found', status = 500, 'Internal Server Error')	StormEvents |project EndTime, StartTime |extend Duration = EndTime - StartTime)	|Event.Rule=330009.2 |eval state= if (Data.Exception = "0", "success", "error")

AQL Operators by Use Case

Use Case		AQL Operators
SOAR	ThreatStream Enrichments	"abuseipdb" on page 428, "domaintools" on page 525, "openports" on page 622, "recordedfuture" on page 633, "mdti_ssl" on page 613, "shodan_database" on page 652, "spath" on page 661, "virustotalv3" on page 699, "whois" on page 712
	Investigations	"creinv" on page 507, "jira" on page 587, "snow" on page 654
	Preferred Tags	"lstperftag" on page 605, "addtag" on page 435, "rmvtag" on page 641
	Other ThreatStream	"tsquery" on page 696, "sandbox" on page 642

Use Case		AQL Operators
UEIBA	Model Management	"fit" on page 539, "apply" on page 451, "listmodels" on page 598, "ml_model_summary" on page 614, "deletemodel" on page 524
	Algorithms and Functions	"aggr" on page 436, "fit ARIMA" on page 542, "fit birch" on page 546, "fit DBSCAN" on page 547, "fit DecisionTreeRegressor" on page 548, "fit LinearRegression" on page 559, "fit LocalOutlierFactor" on page 561, "fit LogisticRegression" on page 563, "fit RandomForest" on page 568, "fit StandardScaler" on page 572, "kmeans" on page 592, "score" on page 644

Operators with Functions

AQL Operator	Functions
"aggr" on page 436	"array_agg()" on page 437, "avg()" on page 437, "count()" on page 437, "count(distinct <field>)" on page 437, "count_distinct()" on page 437, "distinct ()" on page 437, first, last, "max()" on page 438, "median()" on page 438, "min()" on page 438, "mode()" on page 438, "perc()" on page 438, "range ()" on page 438, "sparkline[(<func>(<field>), <timespan>)]" on page 439, "stdev()" on page 439, "stdevp()" on page 440, "sum()" on page 440, "var ()" on page 440

AQL Best Practices

Use the recommendations below to optimize your queries and use less resources for their calculation.

Use `superapi` Efficiently

The `superapi` operator makes one API call per input row, when the call includes the `input_fields` parameter. When queries pipe many rows through endpoints that return large responses, the cumulative response size can exceed the 50 MB limit, causing some rows to be skipped.

When the total exceeds 50 MB, the operator stops making additional API calls. The size check occurs after each response is received.

If a call pushes the cumulative total over the 50 MB limit, that call completes and returns data normally, and only subsequent calls are skipped. Therefore, rows processed before the limit is reached return normally, while rows that would have been processed after the limit is reached receive a warning in their output instead of API results.

To avoid hitting this limit, reduce the scope of your API calls using the following practices:

1. Limit Input Rows

Add `| limit N` before `| superapi` to cap the number of rows that trigger API calls.

Inefficient Query	Recommended Query

`| makeresults count=100`
`| streamstats count AS counter`
`| superapi endpoint='/api/v1/intelligence/?limit=1000&offset=<<counter>>' input_fields='counter'`
	`| makeresults count=100`
	`| streamstats count AS counter`
	`| limit 20`
	`| superapi endpoint='/api/v1/intelligence/?limit=1000&offset=<<counter>>' input_fields='counter'`

2. Use a Smaller Page Size

Lower the `limit=` parameter in your endpoint URL to reduce the size of each individual API response.

Inefficient Query	Recommended Query

`anomali_api_proxy | superapi`
 	
`endpoint='/api/v1/intelligence/?limit=1000'`
	`anomali_api_proxy | superapi`
	
	`endpoint='/api/v1/intelligence/?limit=100'`

3. Filter Before the API Call

Use `| where` conditions before `| superapi` to reduce the number of rows that need API enrichment.

Inefficient Query	Recommended Query

`ocsf`
`| superapi endpoint='/api/v1/actor/<<id>>/tiprepport/'`
`input_fields='id'`
	`ocsf`
	`| where actor_name in ("APT27", "APT28")`
	`| superapi endpoint='/api/v1/actor/<<id>>/tiprepport/'`
	`input_fields='id'`

Use OCSF Index to Reduce Parsing Overhead

If your data exists in OCSF, the OCSF-normalized index pre-parses and structures all event data at ingestion time. Fields are extracted, typed, and stored as structured columns, so queries run directly against pre-processed data with no runtime parsing overhead. By contrast, the `eventlog` index parses raw messages at query time, reducing query speeds and making high-volume ingestion more expensive.

To reduce this parsing overhead, Anomali recommends the following practice when using the `ocsf` Version 1.2 index:

Filter on `category_uid` or `category_name` Early

Always narrow the scope by filtering on `category_uid` or `category_name` early in your query. This allows the query planner to target only the relevant category tables instead of scanning all of them, significantly reducing the amount of data scanned and improving query speed.

Inefficient Query	Recommended Query
`ocsf | where metadata.log_name contains "windows"`	`ocsf | where category_uid in (4, 5) and metadata.log_name contains "windows"`
`ocsf | where src_ip = "192.168.1.1"`	`ocsf | where category_name in ("Network Activity") and src_ip = "192.168.1.1"`

See "OCSF Schema Overview" on page 39 for more information on OCSF Version 1.2 parsing and naming conventions.

Resolving error: "To optimize your experience and reduce your AVC usage, Anomali recommends reducing the time range to 24h to proceed."

When you issue a search, you might encounter the error mentioned above. To resolve this issue, Anomali recommends the following practices:

1. Specify Sourcetype with Parsed Fields:

Use sourcetype as the primary constraint filter in the search query, while using parsed fields. See the Anomali Security Analytics Data Onboarding Guide for the complete list of supported log sources.

Inefficient Query	Recommended Query
`| where message contains_ci '23b6c45a48c45a4c3b56a2c09a'`	`| where sourcetype = 'aws_cloudtrail' and file_hash = '23b6c45a48c45a4c3b56a2c09a'`

2. Avoid Wildcard Matching on Sourcetype

Avoid performing any wildcard matching such as `sourcetype contains '%'` in the search query.

Inefficient Query	Recommended Query
`| where sourcetype contains '%'`	`| where sourcetype = 'defender'`

3. Avoid message on large search ranges

Avoid using the **message** field on large search time ranges. Anomali recommends that you restrict a search query with the `message` field to only 24 hours.

Inefficient Query	Recommended Query
`| where message contains_ci 'RunInstances'`	`| where sourcetype = 'okta' | where message contains_ci 'RunInstances'`

4. Turbosearch on Indexed Fields

Use the turbosearch operator on the following indexed fields:

- `account_id`
- `dest`
- `dest_ip`
- `dvc_name`
- `file_hash`
- `file_hash2`
- `file_hash3`
- `file_name`
- `host`
- `http_referrer`
- `message_id`
- `query`
- `src`
- `src_ip`

- `url`
- `user`

Inefficient Query	Recommended Query

`| where sourcetype = 'aws_cloudtrail'`
`| where message contains '23b6c45a48c45a4c3b56a2c09a'`
`| where event_name = 'RunInstances'`
	`| turbosearch '23b6c45a48c45a4c3b56a2c09a'`
	`| where sourcetype = 'aws_cloudtrail' and event_name = 'RunInstances'`

If you require additional fields from the eventlog schema, Anomali recommends that you contact Anomali Support for assistance in enabling those for your needs.

Searching across different time ranges

Anomali lets you issue searches across multiple time ranges. Anomali recommends the following solutions for launching efficient searches across different time ranges:

I want to search for a time range of	Anomali's Recommendation	Additional Information
Less than 7 days	Use a regular search, considering all the AQL Best Practices. For example: `| where sourcetype = 'aws_cloudtrail' and file_hash = '23b6c45a48c45a4c3b56a2c09a'`	See defining your AQL search query for more information on how to create an AQL query expressions and define a search time range.

7 days to 90 days	Use the turbosearch operator to analyze event data across large time spans, up to 90 days. For example:
	`| turbosearch '23b6c45a48c45a4c3b56a2c09a'`
	`| where sourcetype = 'aws_cloudtrail'`
		See turbosearch.

I want to search for a time range of	Anomali's Recommendation	Additional Information
Greater than 90 days	Your data is archived beyond 90 days. Refer to the Anomali Search Data Retention Policy to understand the next steps to search this data.	See Anomali Virtual Compute to understand its usage and the different user actions that lead to AVC usage.

Use `rex` Operator and `json_extract` Efficiently

To reduce the number of records the `rex` operator and `json_extract` process, Anomali recommends using the `rex` operator and the `calc` operator's `json_extract` function **after** all conditional clauses used to filter the data.

Before	After

`|where sourcetype = 'cisco'`
`|rex field=message '.*?msg=(?<agent_msg>.*?)rawEvent=(?<rawEvent>.*?)$'`
`|where action = 'create'`
`|fields action, event_time, category, sourcetype`
	`|where sourcetype = 'cisco' AND action = 'create'`
	`|rex field=message '.*?msg=(?<agent_msg>.*?)rawEvent=(?<rawEvent>.*?)$'`
	`|fields action, event_time, agent_msg, rawEvent`

Use `array_agg` Operator Effectively

The `array_agg` operator enforces a limit of 50 distinct values. If the data being aggregated exceeds this limit, the results are truncated. Anomali recommends the following solutions to effectively use `array_agg` and avoid hitting resource limits:

- Use low cardinality fields.
- Filter or transform the data before applying the `array_agg` operator.
- Use the group by clause for high cardinality fields.

Use the `dedup` Operator Effectively

- Use the `dedup` operator over small data sets. Reduce the data set size by limiting time range or filtering out events.

- Do not apply the `dedup` operator to the `message` field. Doing so impacts search performance because each raw event is held in memory for this operation.

Stagger Scheduled Tasks

Stagger any task that you can schedule to avoid overloading the system with too many scheduled jobs. For example, instead of scheduling lots of alerts at 2:00 AM, consider staggering them between 2:00 AM and 3:00 AM.

> **Note:** Even if you schedule a job at, say, 2:00 AM, the system might stagger it to balance workloads.

Anomali recommends you consider staggering the following tasks:

- Scheduled Alerts
- Scheduled Reports

Use Definitive Operators

If possible, use definitive operators such as `=` and `!=` instead of operators such as `contains`. For more accurate results, check the values in your tables before using them in the query.

Before	After
`|where sourcetype contains_ci "defender"`	`|where sourcetype = "ms_defender"`

Place Limits Last

Place the `limit` operator last in the query to avoid getting non-deterministic results. In the Before example, the `limit` is placed before the second `where` clause, which means that the `where` clause will be applied only to 100000 events.

If you place the `limit` operator last, all matching events are calculated first and then limited.

Before	After

`|where sourcetype contains_ci "okta"`
`|calc dtHash= json_extract(message, 'debugContext.debugData.dtHash')`
`|aggr count(message) by dtHash`
`|limit 100000`
`|where dthash="<hash>"`
	`|where sourcetype contains_ci "okta"`
	`|calc dtHash= json_extract(message, 'debugContext.debugData.dtHash')`
	`|aggr count(message) by dtHash`
	`|where dthash="<hash>"`
	`|limit 100000`

Use Smallest Possible Time Range

Limit your query to the shortest time frame necessary. The larger the time range, the more data that needs to be scanned, which increases the query execution time.

Specify Targeted Search Table

Specify a system lookup table, such as `iocmatch`, or a custom lookup table to reduce the data volume scanned. By default, queries scan the `eventlog` table, which contains all organizational data.

Example
`iocmatch | where action in ('allow', 'accept')`

Prioritize Predicates

Prioritize predicates (`where` statements) in terms of order: place them earlier but only if it does not affect the accuracy of the query.

In the Before example, the `calc` line uses the `bytes_in` and `bytes_out` fields, which are unrelated to the `where` clause, which uses `sourcetype`. You can place the `where` clause before, and now calculations will be skipped for fields with the source type other than `aws_cloudtrail`.

Before	After
`|calc total_bytes = bytes_in + bytes_out` `|where sourcetype = "aws_cloudtrail"`	`|where sourcetype = "aws_cloudtrail"` `|calc total_bytes = bytes_in + bytes_out`

Use Normalized Fields

When extracting values from JSON, check if the field to extract is a normalized column in the eventlog table schema. If yes, use the normalized column instead.

In the After example, the `username` field in the JSON string message field has been normalized as `user` in the table schema.

Before	After
`|calc username = json_extract (message, "username")` `|fields username`	`|fields user`

Dotted notation

Use the dotted notation when handling a JSON field in order to expedite the use of it with different operators.

Anomali Search supports dotted notation for both eventlog schema fields (extracted from JSON messages) and OCSF Version 1.2 schema fields.

OCSF uses dotted notation natively for nested object structures. For example, OCSF fields like `src_endpoint.ip`, `actor.user.name`, and `dst_endpoint.port` use dotted notation to represent hierarchical data. See "OCSF Schema Overview" on page 39 for more information on OCSF field naming conventions.

Syntax

To access a JSON array using notation, use the following syntax:

```
<json_field_name>.<json_path>
<json_field_name>.<json_path>.<json_array_element>{0}
```

`<json_field_name>`	Name of the table field containing the JSON data.

`.<json_path>`	Specifies the JSON path of the attribute to be used in the operator. To specify any nested property, a dot (`".`")must be used. For JSON data that is extracted using the `{}` brackets, this will automatically convert the output type to `ARRAY`. The output can be used directly with array functions like `array_contains`. For more information about the syntax used in `json_path`, please refer to the `calc json_extract` documentation.
`<json_array_element>`	The child element in the array.
`{0}`	`{0}` indicates the position of the element to access in the JSON array, similar to `json_extract`. Supports positive integers, including zero, as positional values, for example, `{0}`, `{9}`, `{15}`. Variables are not allowed. For more information about the syntax used in `json_path`, please refer to the `json_extract` documentation.

Usage

You can use the dotted notation to perform any operation on the JSON fields, in conjunction with other operators as per need. JSON keys can be nested.

Common Event Format logs

In the case of Common Event Format (CEF) logs, the following rules apply:

- Only the first CEF log is parsed.
- Must be named using dotted notation, such as:
  - Example 1: `message.cef.version`
  - Example 2: `message.cef.deviceVendor`
- The following fields are supported for CEF format:
  - `version`
  - `deviceVendor`

- `deviceProduct`
- `signature`
- `name`
- `severity`

Key-value pairs:

In the case of key-value pairs, the following rules apply:

- Pairs can contain equal sign ("=")
- Empty value in a key-value pair will be handled as an empty string
- Pairs can be line-separated or space-separated
- Pairs do not support nesting.

Examples

The following examples demonstrate dotted notation usage with both eventlog JSON extraction and OCSF Version 1.2 fields. For eventlog data, dotted notation extracts nested values from JSON messages on-the-fly. For OCSF data, dotted notation is the native field naming convention for hierarchical structures. Both approaches allow you to access nested data using the same familiar syntax.

Eventlog JSON Extraction

On the Anomali Search platform, here is an example of a typical raw message (in JSON format) from Okta logs whose fields can be extracted on-the-fly.

```
message                 ▾Object
                          ▾actor: Object
                            id: "00uttidj01jqL21aM1d6"
                            type: "User"
                            alternateId: "jane.smith@acme.com"
                            displayName: "Jane Smith"
                            detailEntry: null
                          ▾client: Object
                            ▾userAgent: Object
                              rawUserAgent: "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWe
                              bKit/537.36 (KHTML, like Gecko) Chrome/127.0.0.0 Safari/537.36"
                              os: "Mac OS X"
                              browser: "CHROME"
                            zone: null
                            device: "Computer"
                            id: null
                            ipAddress: "10.100.0.54"
                            ▾geographicalContext: Object
                              city: "New York"
                              state: "New York"
                              country: "United States"
                              postalCode: 10013
                              ▾geolocation: Object
                                lat: 40.3157
                                lon: -74.01
                          ▸device: Object
                          ▾authenticationContext: Object
                            authenticationProvider: null
                            credentialProvider: null
                            credentialType: null
                            issuer: null
                            interface: null
                            authenticationStep: 0
                            rootSessionId: "idxBager62CSveUkTxvgRtonA"
                            externalSessionId: "idxBager62CSveUkTxvgRtonA"
```


Field examples:

Example 1: Extract nested object fields using dot notation.

```
message.actor.id
```

Example 2: Access array elements with bracket notation.

```
message.client.userAgent.browser
```

Example 3: Navigate complex nested data structures.

```
message.request.ipChain{0}.geographicalContext.postalCode
```

Example queries with other operators:

Example 1: Filter Okta logs where the client device is a computer.

```
eventlog | where sourcetype = "okta" | where
message.client.device = "Computer"
```

Example 2: Filter Okta logs by client geographical context country.

```
eventlog | where sourcetype = "okta" | where
message.client.geographicalContext.country = "United States"
```

Example 3: Filter Office 365 logs by risk event type array element.

```
eventlog | where sourcetype = "office365" | where
message.riskEventTypes{0} = "unfamiliarLocation"
```

OCSF Version 1.2 Fields

Using the OCSF Version 1.2 field conventions, you can create queries using OCSF fields with dotted notation.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Example queries with OCSF fields:

Example 1: Filter OCSF Network Activity events by source IP address.

```
ocsf | where category_name = "Network Activity" and src_
endpoint.ip = "1.2.3.4"
```

Example 2: Filter OCSF Authentication events by user name.

```
ocsf | where class_name = "Authentication" and actor.user.name =
"example_admin"
```

Example 3: Filter OCSF Security Finding events by severity and product name.

```
ocsf | where class_name = "Security Finding" and severity_id = 3
and metadata.product.name = "CrowdStrike"
```

Subsearch

Anomali Search lets you embed a search query within the primary search and use the results of this embedded query as parameters within the primary search, enabling you to create complex search queries and perform correlation across multiple datasets.

With the ability to filter, manipulate, and transform results within the primary search, subsearches let you combine multiple search queries, yield the results, and join them using one single query, leading to easier, faster, and more efficient understanding and use of Search.

Additionally, the search bar displays auto-complete suggestions to help you build a valid search query, simplifying the search experience.

Benefits

By using subsearches, you can:

- Simplify search queries by embedding multiple queries within the same primary search.
- Create more powerful queries that dynamically adapt according to the results of the embedded query.
- Enhance the quality of threat hunting by efficiently querying multiple datasets simultaneously.
- Reduce workload by eliminating the need to create multiple queries.

How does it work?

Use square brackets (`[]`) to embed a embedded query within the primary search query and use the results for further analyses.

```
| where sourcetype = "aws_vpc_flow"
```

```
| join left [
    observables
    | where ioc_type = "ip" and confidence > 75 and severity != "low"
]
on dest_ip = indicator output indicator, confidence, severity, feed_id, tags
```

In the example above, the subsearch captures IP addresses for AWS VPC Flow logs that have high severity with a confidence score > 75.

Key Highlights

- You can define up to 6 subsearches in one primary search query.
- You cannot perform subsearch nesting, that is, you cannot embed one subsearch inside another subsearch.
- A subsearch will timeout after 180 seconds.
- Only the first 10000 result rows are considered as the output from a subsearch, which are then passed on to the primary search query.
- You must provide as output only one field with the `fields` operator, if you use the `where` and `in` operators in a subsearch filter.
- You can define a search time range that is different from the range of the primary search by using the `earliest | latest` function within the where operator.
  - Example: `| aggr count by sourcetype | appendtable [| where earliest=10d and latest=5d | aggr count by sourcetype]`
- When used with `appendtable`, a subsearch will fail if the primary search does not have any data. However, a regular primary search with an empty subsearch will function correctly.

> **Caution:** This is currently a known issue. A resolution will be deployed in an upcoming release.

Use Case

Filtering

Filter the primary search results by defining conditional filters within the embedded search. The following subsearch returns the top 10 users that encountered a failed login attempt using the top operator as its conditional filter.

> **Note:** You must provide as output only one field with the `fields`, if you use the `where` and `in` operators in a subsearch filter.

```
| where event_name contains_ci "userlogonfailed"
| where user in [
    | where event_name contains_ci "userlogonfailed"
    | top 10 user
    | fields user
]
```

Using OCSF Fields in Subsearches

Subsearches support OCSF Version 1.2 fields with dotted notation. Field references from parent queries are properly resolved in subsearch contexts, allowing you to construct complex queries on OCSF fields.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Example 1: Join with subsearch using OCSF fields

```
ocsf
```

```
| fields metadata.product.name, activity_name
| dedup metadata.product.name
| join left [
    ocsf
    | aggr count as cnt by metadata.product.name
] on metadata.product.name=metadata.product.name output cnt
```

Example 2: Join with observables using subsearch on OCSF endpoint field

```
ocsf
| fields metadata.product.name, activity_name , dst_endpoint.ip
| join left [
  observables
  | where ioc_type = 'ip' and confidence > 75 and severity!='low'
  | dedup indicator
] on dst_endpoint.ip = indicator output indicator, confidence, severity, feed_id, tags
```

Example 3: Nested subsearches with IN operator using OCSF user fields

```
ocsf
| where metadata.product.name!='unknown_source' and isnotnull(activity_name) and (actor.user.name in
  [ocsf
```

```
    | where isnotnull(actor.user.name) and activity_name contains_ci 'Success'
    | top 2 actor.user.name | fields actor.user.name
  ]
OR actor.user.name in
  [ocsf
    | where isnotnull(actor.user.name) and activity_name contains_ci 'create'
    | aggr count as cnt by actor.user.name
    | sort cnt desc
    | head 2
    | fields actor.user.name
  ])
| fields activity_name, actor.user.name
```

Generation

Generate values to be used by parameters in the primary search. In the following example, the subsearch generates a list of IP addresses with high severity and more than 75% confidence, which is then joined with destination IP addresses to obtain the final result.

```
| where sourcetype = "aws_vpc_flow"
| join left [
    observables
    | where ioc_type = "ip" and confidence > 75 and severity != "low"
]
```

```
on dest_ip = indicator output indicator, confidence, severity,
feed_id, tags
```

Manipulation

Append results from different searches into the primary search. In the example below, the subsearch fetches the command string of the parent process and appends them to the eventlog table.

```
eventlog | appendcols [iocmatch | field parent_command_line]
```

Operator Scope

Subsearches allow you to use all AQL Search operators. Once a subsearch runs successfully, its results are passed on as parameters to the primary search.

Primary searches that involve embedded queries support the following operators:

- where
- join
- appendtable
- appendcols
- set
- turbosearch

Subsearch Audit

You can obtain audit logs for subsearch using the `audit` keyword, one log for each subsearch and one entry for the primary search. In the following example, the user `yourusername` issues a search with a subsearch, and then obtains audit logs accordingly.

```
audit

| where user contains 'yourusername'
```

```
| fields action, search_job_id, search_query, response_digest
```

abuseipdb

Use the `abuseipdb` operator to enrich results related to an IP address or field containing an IP address using the ThreatStream AubseIPDB enrichment.

Syntax

`|abuseipdb <field or value>`

`<field or value>`	Specify the field name or IP address to look up.

Usage

You can use the `abuseipdb` operator to enrich results that inform your security orchestration, automation and response actions. A request to the AbuseIPDB enrichment returns columns of data similar to the following.

ip	isPublic	isWhitelisted	abuseConfidenceScore	usageType	domain	isTor	totalReports	lastReportedAt
'7.7.7.7'	True	False	0	"Military"	"nic.mil"	False	0	2022-8-12 12-1-5

Examples

Example 1: Enrich IP address fields with AbuseIPDB threat data.

```
|fields ip |abuseipdb(ip)
```

Example 2: Retrieve AbuseIPDB reports for a specified IP address.

```
|fields src_ip |abuseipdb(src_ip)
```

accum

Use the `accum` operator to get the running total or sum of a numeric field.

Syntax

```
|accum <field> [as <new_field>]
```

`<field>`	Required. Specify the numeric field to accumulate.
`as <new_field>`	Optional. Specify a column name for the calculated accumulation.

Usage

You can use the `accum` operator to get the running total or sum of a numeric field. Optionally, you can specify a new field name for the calculated sum. If you do not specify this option, the new column name follows the pattern `accum_<field>`.

Examples

Example 1: Calculate running total of a volume field.

```
|accum volume
```

Example 2: Create a named column for the running accumulation.

```
|accum volume as running_vol
```

addcoltotals

Use the `addcoltotals` operator to add up all numeric fields in a results set or a selection of fields in a results set.

Syntax

```
|addcoltotals [<fields>] [labelfield = "<string>"]
[label = "<string>"]
```

`<fields>`	Optional. Comma-separated list of field names to be summed. If not provided, the operator sums all numeric fields.
`labelfield = "<string>"`	Optional. Specify a field name to add to the results set.
`label = "<string>"`	Optional. Add a label to the result set totals. If the `labelfield` option is not set, this argument has no effect.

Usage

You can use the `addcoltotals` operator to quickly summarize numeric fields in your result set without having to calculate each one individually.

Examples

Example 1: Sum all numeric fields in the results.

```
|addcoltotals
```

Example 2: Sum specific numeric columns.

```
|addcoltotals num_incidents, num_views
```

Example 3: Add labeled totals row to the results table.

```
|addcoltotals labelfield = "totalfield" label = "Total"
```

addinfo

Use the `addinfo` operator to add metadata about the search to search results.

Syntax

`|addinfo`

Usage

The `addinfo` operator adds the following metadata fields to the search result.

info_min_time	Start time of a search query time range (Unix timestamp).
info_max_time	End time of a search query time range (Unix timestamp).
info_sid	Search job ID.
info_search_time	Human readable search date and time.

Examples

Example 1: Add search metadata fields to aggregated results.

```
|aggr count by src |addinfo
```

Example 2: Retrieves metadata about the current search execution for the first 5 OCSF events, surfacing the search time range boundaries, session ID, and when the search ran.

```
ocsf | addinfo | limit 5 | fields info_min_time, info_max_time,
info_sid, info_search_time
```

addinv

Use the `addinv` operator to add elements to an existing ThreatStream investigation.

Syntax

```
|addinv investigation_id=<int> [r_id=<int> r_
type=<string> | fields=<string>]
```

`investigation_id=<int>`	Required. Specify the ID of the investigation to which you need to add elements.
`r_id=<int>`	Required for adding a single element. Specify the ID of the element that will be added to the investigation.
`r_type=<string>`	Required for adding a single element. Specify the type of the element that will be added to the investigation. Possible options: • `actor`: Actor • `campaign`: Campaign • `incident`: Incident • `intelligence2`: Observable • `signature`: Signature • `tipreport`: Threat Bulletin • `ttp`: TTP • `vulnerability`: Vulnerability

`fields=<string>`	To add elements from in a previous table, specify a comma-separated list of columns that contain the required fields for the elements. The list must not have spaces between elements. • If the previous table already has the `r_id` and `r_type` columns, then specify `fields=r_id,r_type` • If the previous table has the values in the columns named differently, then specify `fields='<column1> as r_id,<column2> as r_type'` **Notes**: • The column used as `r_id` must be numeric. • The column used as `r_type` must contain the values mentioned in the `r_type` argument.

Usage

You can use the `addinv` operator to add a single element to an investigation. It also supports the table mode, which uses the values from a previous table to add one element per row to an investigation.

Examples

Example 1: Add a single TTP element to an investigation.

```
addinv_test_table | addinv investigation_id=33 r_id=7 r_
type='ttp'
```

Example 2: Add multiple elements from a table with matching column names.

```
addinv_test_table | addinv investigation_id=33 fields=r_id,r_type
```

Example 3: Add elements using aliased column names.

```
addinv_test_table | addinv investigation_id=33 fields='element_id
as r_id,type as r_type'
```

Example 4: Add observable elements from search results to an investigation.

```
eventlog | where sourcetype = "okta" | addinv investigation_id=33
fields='id as r_id,type as r_type'
```

addsums

Use the `addsums` (also called `addtotals`) operator to compute the row sum of specified columns as a new row.

Syntax

`|addsums [<fields>] [fieldname=<str>]`

`|addtotals [<fields>] [fieldname=<str>]`

`<fields>`	Optional. Specify a comma-separated list of field names to be summed. If not provided, the operator sums all numeric fields.
`fieldname=<str>`	Optional. Specify the output field name. If the option is not specified, a field named `total` is added to results.

Usage

You can use the `addsums` operator to create a new row that has the sum of values for the specified fields.

Examples

Example 1: Calculate row sums for columns `val1, val2, val3` and assigns it to a new column called `Totals`.

```
|addsums val1,val2,val3 fieldname=Totals
```

Example 2: Filters OCSF events to those with valid src and dst ports, sums the port numbers per user, then appends the Total row aggregating the grand sum of `total_src_port` and `total_dest_port` across all users.

```
ocsf | where src_endpoint.port>0 and dst_endpoint.port>0 | aggr
sum(src_endpoint.port) as total_src_port, sum(dst_endpoint.port)
as total_dest_port by actor.user.name | addsums total_src_port,
total_dest_port fieldname=Total
```

addtag

Use the `tag` operator to add a tag or multiple tags to ThreatStream observables for the IOCs returned by an ioc or iocmatch lookup table search.

Syntax

`|addtag <tagnames> [is_public=true|false]`

`<tagnames>`	Required. Specify a comma-separated list of tag names to add to the IOCs.
`is_public`	Optional. Specify `true` to set the added tag public setting to true. If this option is not specified, the implicit default is `false`: the tag is not publicly visible.

Usage

You can use the `addtag` operator to add a tag or multiple tags to ThreatStream observables for the IOCs returned by an ioc or iocmatch lookup table search. For example, you can perform a search on the ioc lookup table to find indicators with a confidence score of greater than 80 and high severity; and then pipe this operator to add a specific tag the organization uses to identify indicators that are of high concern and routinely sent to Integrator destinations.

In addition to adding the tag in ThreatStream, the operator adds to search results a column named `tag_added` with the value `True`.

For information about ThreatStream observable tags, refer to ThreatStream online help.

Examples

Example 1: Add a single tag to observables.

```
|addtag "tag_name_1"
```

Example 2: Add multiple tags to observables.

```
|addtag "tag_name_1, tag_name_1"
```

Example 3: Add a public tag to observables.

```
|addtag "tag_name_1" is_public=true
```

Example 4: Tag filtered IOCs from a lookup table based on confidence score.

```
iocmatch |where not feed_name contains "OSINT" and confidence >80
and ioc_type!="md5" and indicator startswith "50"|dedup id
|fields id, indicator, feed_name, tags |addtag "ioctag_test"
```

aggr

Use aggregation functions to summarize results as a single value (or an array).

You can also drilldown on the aggregated values to launch a new search that uses these aggregated values as a filter in the `where` operator and display more details about these specific values. See Aggregate, Sort, and Limit Result Rows for more information.

Syntax

```
|aggr <aggr function> [as new_field] [, <aggr
function> [as new_field], ...] by <field> [,
<field>, ...]
```

Usage

Use `aggr` to drilldown on the aggregated values to launch a new search that uses these aggregated values as a filter in the `where` operator and display more details about these specific values. See Aggregate, Sort, and Limit Result Rows for more information.

`aggr` supports the following aggregation functions:

Function	Usage
`array_agg()` `values()`	Return an array of distinct result values. The `array_agg()` and `values()` functions are equivalent. **Note:** This function consumes significant memory to store aggregated values. See AQL Best Practices to use `array_agg()` effectively.
`avg()`	Return the average of result values. Applies to numeric and date fields.
`count()`	Return a count of results.
`count(distinct <field>)`	Return a count of distinct values in results.

`count_distinct()`
`distinct_count()`
`dc()`
	Return a count of distinct values in results.

`distinct()`	Returns distinct values for the specified field.

Function	Usage
`first()`	Returns the first entry in a result set. For more information, see first.
`last()`	Returns the last entry in a result set. For more information, see last.
`latest()`	Returns the latest or the most recent entry in a result set.
`max()`	Return the highest value among results. Applies to numeric and date fields.
`median()`	Return the middle value of ordered results. Applies to numeric and date fields.
`min()`	Return the lowest value among results. Applies to numeric and date fields.
`mode()`	Return the most frequent value of the specified field. Applies to numeric and date fields.

`perc()`	Return the specified percentile value of the specified field. Applies to numeric fields only.
	The function has the following syntax:
	`perc(<field>, <percentile>)`
	The following example returns the 25th percentile value for the volume field:
	`|aggr perc(volume, 25)`

`range()`	Return the difference between the lowest and highest value for the specified field. Applies to numeric fields.

Function	Usage
`sparkline[(<func>(<field>), <timespan>)]`	Returns one or more columns having a series of computed values to plot a sparkline visualization. The maximum number of buckets is 60. If you do not specify options, the default aggregation function is `count(*)` and the timespan is computed dynamically. You can specify one of the following options: • `<func>(<field>)`: Specifies an aggregation function and field argument. • `<timespan>`: Specifies a timespan for buckets • Specify timespan in the the following form: `<integer>{s,h,m,d,w,mon,y}` • Multiple units are not allowed for week, month, or year. In other words, specify `1` for the integer place when specifying week, month, or year. • Similar to the timetrend operator, after you enter a search, the value for timespan may be recalculated and rationalized to create meaningful buckets, given the configured time range.
`stdev()`	Return the sample standard deviation for the set of values returned the specified field.

Function	Usage
`stdevp()`	Return the population standard deviation for the set of values returned the specified field.
`sum()`	Return the sum of result values. Applies to numeric and date fields.
`var()`	Return the sample variance of a data set. Applies to numeric and date fields. Example: `|aggr var(field) as field_variance` **Note**: In a complex AQL expression that includes `aggr sum()` operations, the variance function should be last in the sequence.

Examples

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Example 1: Aggregate distinct source port values grouped by source port.

```
|aggr array_agg(src_port) by src_port
```

Example 2: Calculate the average of incoming bytes.

```
|aggr avg(bytes_in)
```

Example 3: Count events grouped by source address.

```
|aggr count by src
```

Example 4: Count distinct source addresses with a custom result field name.

```
|aggr count(distinct src) as distinct_sources
```

Example 5: Find the highest destination port value.

```
|aggr max(dest_port)
```

Example 6: Calculate the range of byte values.

```
|aggr range(bytes_in)
```

Example 7: Calculate standard deviation of an age field.

```
|aggr stdev(age)
```

Example 8: Visualize data with sparklines grouped by source type.

```
|aggr sparkline, count as cnt by sourcetype
```

Example 10: Counts the number of unique source IPs per OCSF event category.

```
ocsf | aggr count(distinct src_endpoint.ip) by category_uid
```

Example 11: Generates a sparkline of hourly OCSF event counts per product name, visualizing activity trends for each security source in a compact inline chart.

```
ocsf | aggr sparkline(count(class_name), 1h) as sparkline_action
by metadata.product.name
```

anomalousvalue

Use the `anomalousvalue` operator to compute an anomaly score for each field of each event, relative to the values of this field across other events.

Syntax

```
|anomalousvalue <av-options>
[action=annotate|filter|summary] [pthresh=<number>]
[<field-list>]
```

`<av-options>`	Required. Specify one or more option to control which fields are considered for discriminating anomalies. Possible options: • `maxanofreq=<float>`: Maximum anomalous frequency is expressed as a floating point number between 0 and 1. Omits a field from consideration if the field is too frequently anomalous. If the ratio of anomalous occurrences of the field to the total number of occurrences of the field is greater than the `maxanofreq` value, then the field is removed from consideration. Default: `0.05` • `minnormfreq=<float>` : Minimum normal frequency is expressed as a floating point number between 0 and 1. Omits a field from consideration if the field is not anomalous frequently enough. If the ratio of anomalous occurrences of the field to the total number of occurrences of the field is smaller than p, then the field is removed from consideration. Default: `0.01`

	• `minsupcount=<int>` : Minimum supported count must be a positive integer. Drops a field that has a small number of occurrences in the input result set. If the field appears fewer than N times in the input events, the field is removed from consideration. Default: `100` • `minsupfreq=<float>`: Minimum supported frequency is expressed as a floating point number between 0 and 1. Drops a field that has a low frequency of occurrence. The `minsupfreq` argument checks the ratio of occurrences of the field to the total number of events. If this ratio is smaller than p, the field is removed from consideration. Default: `0.05`

`action=annotate|filter|summary`	Specify whether to return the anomaly score (annotate), filter out events that are not anomalous values (filter), or return a summary of anomaly statistics (summary). Possible options: • `annotate`: Adds new fields to the events containing anomalous values. • `filter`: Returns events with anomalous values. Events without anomalous values are removed. The events that are returned are annotated, as described for `annotate`. • `summary`: Returns a table summarizing the anomaly statistics for each generated field. The table includes the number of events that contain this field, the fraction of events that are anomalous, what type of test (categorical or numerical) have been performed, and so on. Default: `filter`
`pthresh=<number>`	Optional. Specify a probability threshold (as a decimal) that has to be met for a value to be considered anomalous.

`<field-list>`	Optional. Specify a comma-separated or space-separated list of fields that you need to analyze. Default: empty If the field list is not specified, all fields are considered.

Usage

You can use the `anomalousvalue` operator to detect outliers or anomalous values in a dataset. This can be particularly useful for identifying unusual patterns or detecting potential issues in your data. The operator applies statistical methods to determine which values deviate significantly from the expected range based on historical data.

For numerical fields, `anomalousvalue` identifies or summarizes the values in the data that are anomalous either by frequency of occurrence or number of standard deviations from the mean. For fields that are determined to be anomalous, a new field is added according to the selected action.

Examples

Example 1: Filter events with anomalous values below a probability threshold.

```
|anomalousvalue action=filter pthresh=0.01
```

Example 2: Detect and annotate anomalous values in the dataset.

```
|anomalousvalue action=annotate
```

anomalydetection

Use the `anomalydetection` operator to find outliers in values of numeric fields.

Syntax

```
|anomalydetection method="iqr" [anomalydet_action=
{"transform"|"remove"}] [<fields>]
```

`method="iqr"`	Required. Specify the detection method. The current available option: `"iqr"`.
`anomalydet_action={"transform"|"remove"}`	Optional. Specify the action to be taken with the detected outliers. Possible options: • `"transform"`: Transforms the fields based on the outliers. • `"remove"`: Removes the outlier values. If you specify the `anomalydet_action` attribute, the `anomalydetection` operator will work the same as the "outlier" on page 622 operator.
`<fields>`	Optional. Specify a comma-separated or space-separated list of numeric fields to check for outliers.

Usage

You can use the `anomalydetection` operator to find anomalies in numeric fields and, if needed, modify them. Currently, this operator supports only the IQR method.

Examples

Example 1: Detect outliers using the interquartile range method.

```
|anomalydetection method="iqr"
```

Example 2: Transform detected outlier values to boundaries.

```
|anomalydetection method="iqr" anomalydet_action="transform"
```

Example 3: Remove outlier values from the results.

```
|anomalydetection method="iqr" anomalydet_action
```

appendcols

Use the `appendcols` operator to define a subsearch to append the output columns to the results of the main search, or conditionally replace the column values with the subsearch output values.

Syntax

```
| appendcols [override=true|false] [<subsearch>]
```

`[override=true|false]`	Specify `true` to replace the main query result field values with the subsearch result field values in the search results table (if the same fields appear in results for both queries). When the override option is used, the results table has the following combination: • the fields from the main query, which are not present in the subsearch results. • the fields which are present in both main and subsearch results, and subsearch results are used. • the fields which are present only in the subsearch results. The implicit default is false. The subsearch results are appended to the results of the main query. If the main and subsearch share common fields, and the result set of the subsearch is larger than the main results, the difference of the subsearch results is added to the main results.
`[<subsearch>]`	Specify an AQL subsearch in square brackets.

Usage

A main query comprises the piped syntax that precedes the `appendcols` piped operator.

A subsearch is the query following the `appendcols` piped operator.

The search results combine the two queries. If the results have any of the same fields, you can use the `override` option to replace the main query result field values with the subsearch result field values.

Examples

Example 1: Replace field values using subsearch results.

The following AQL query uses the override option to replace field values returned in the main query (in this case, the default search) with the values returned by the `where` operator specified in the subsearch:

```
| appendcols override=true [| where action='Allowed' and src_
port=1122]
```

Example 2: Append new columns from a subsearch calculation.

The following AQL query does not use the override option. The subsearch adds a new field created by the calc function:

```
| limit 5 | appendcols override  [| limit 1 | calc new_
field="test"]
```

appendtable

Use the `appendtable` operator to perform a subsearch and append the results from both the searches together.

See Subsearch documentation for more information on how you can create more powerful queries that dynamically adapt according to the results of the embedded query.

Syntax

`|appendtable [eventlog | fields src_port, dest_port]`

`[subsearch]`	A secondary search where you specify the source of the events that you want to append. The subsearch must be enclosed in square brackets (`[]`).

Usage

The `appendtable` is a transformation operator. You can use it to view the results of a subsearch together with the existing results.

Examples

Example 1: Append computed field results to main query results.

```
|appendtable_test | fields src_port, dest_port, action |
appendtable [appendtable_test | calc x = 123 ] | sort x
```

Example 2: Append subsearch results with a different field set.

```
| fields src, dest | appendtable [| fields sourcetype]
```

Example 3: Append filtered subsearch results to main results.

```
| fields src, dest | appendtable [| where src != ‘test’]
```

Example 4: Append calculation results from subsearch.

```
| fields src, dest | appendtable [| calc x=’test’]
```

apply

Use the `apply` operator to compute predictions for the current search results based on a model that was learned using the `fit` operator.

Support for explicit arguments

The `apply` operator supports explicit feature arguments, ensuring increased context and, therefore, better analysis between original and predicted fields.

When you specify features after the model name:

- Only those fields are used as the explanatory features for the model.
- All input columns, including context fields, are preserved in the output.
- The output includes both the original fields and the predicted or scored fields, making it easy to analyze anomalies in full context.

When you do not specify features:

- If you do not specify feature fields after `apply`, the operator passes all fields in the pipeline to the operator.
- `apply` keeps all other columns unchanged as part of the output.

Syntax

Syntax 1: `|apply <model_name> [as <alias>] [fields]`

Syntax 2 (explicit feature selection): `|apply <model_name> [as <alias>] [field1] [field2] [field3]...`

`<model_name>`	Required. Specify the name of the model name that was trained and saved using the `fit` operator.
`[as <alias>]`	Optional. Specify a new field name for the result field having the predicted value.

`[fields]`	Optional. Specify one or more fields that can be applied to the model. **Note:** For the following models, you must include the explanatory fields in `[fields]` as features: • Linear Regression • Decision Tree Regressor • Logistic Regression • Local Outlier Detection • Birch For these models, in the syntax above, fields `field1`, `field2`, `field3` represent the explanatory features. However, for models created with Standard Scaler, `[fields]` captures the columns that are to be transformed or scaled. In this case, fields `field1`, `field2`, `field3` represent the feature columns to transform.
`[field1] [field2] [field3]...`	Required. See "Support for explicit arguments" on the previous page for more information.

Usage

You can use the `apply` operator to compute predictions for the current search results based on a model that was learned using the `fit` operator. The `apply` operator must include one or more explanatory fields (independent variables).

The output will include all the input fields and the predicted field(s), helping you analyze the prediction results. Additionally, you can use the `ml_model_summary` operator to store these explanatory fields when creating the model for future reference.

`apply` also supports explicit feature field selection. See "Support for explicit arguments" on page 451 for more information.

Examples

Example 1: Training with Context, Applying with Context

In the following example, you train a Birch clustering model on authentication behavior.

Training a model

Here, `date_day_num` and `date_hour` are used features, while also keeping `src_ip` and `sourcetype` in the training data:

```
calc date_hour = strftime(event_time, "%H")

calc date_hour = tonumber(date_hour)

calc date_day = strftime(event_time, "%a")

calc date_day_num = case(

  date_day = "Sun", 0,

  date_day = "Mon", 1,

  date_day = "Tue", 2,

  date_day = "Wed", 3,

  date_day = "Thu", 4,

  date_day = "Fri", 5,

  date_day = "Sat", 6

)

aggr count as login_count by src_ip, sourcetype, date_day_num,
date_hour

fit Birch login_count from date_day_num date_hour into "birch_
with_context"
```

Applying the model while preserving context

After training, you want to apply the same model and keep `src_ip` and sourcetype in the output, along with the model scores:

```
calc date_hour = strftime(event_time, "%H")

calc date_hour = tonumber(date_hour)

calc date_day = strftime(event_time, "%a")

calc date_day_num = case(

  date_day = "Sun", 0,

  date_day = "Mon", 1,

  date_day = "Tue", 2,

  date_day = "Wed", 3,

  date_day = "Thu", 4,

  date_day = "Fri", 5,

  date_day = "Sat", 6

)

aggr count as login_count by src_ip, sourcetype, date_day_num,
date_hour

apply birch_with_context date_day_num date_hour
```

In the apply phase:

- `date_day_num` and `date_hour` are explicitly passed as features to the `birch_with_context` model.
- `src_ip`, `sourcetype`, and `login_count` remain in the result as context columns.
- The output includes all of the above plus the model's output fields (for example, cluster assignment, anomaly score).

Example 2: Anomalous Authentication Detection with Full Context

This example shows a typical workflow where you:

- Aggregate authentication events.
- Compute behavioral features.
- Apply a saved ML model.
- Investigate anomalies with full user and device context.

Step 1–2: Aggregate and build features

```
aggr count as total_logins,
     sum(failure_count) as failure_count,
     dc(src_ip) as unique_ips,
     dc(country) as unique_countries,
     sum(weekend_logins) as weekend_logins,
     sum(night_logins) as night_logins,
     avg(ti_score) as avg_ti_score
  by user, device_id, time_bucket
calc failure_rate = failure_count / total_logins
calc weekend_rate = weekend_logins / total_logins
calc night_rate = night_logins / total_logins
```

Step 3: Apply the model with explicit features

Instead of stripping context columns with `fields`, call `apply` with the feature list:

```
apply azuread_scaler_jan26
      total_logins
      failure_rate
      unique_ips
      unique_countries
```

```
        weekend_rate

        night_rate

        avg_ti_score
```

In this example:

- The model uses the 7 listed fields as features.
- Context fields such as `user`, `device_id`, `time_bucket`, `src_ip`, and `country` remain in the pipeline.
- The model outputs additional columns, such as `anomaly_score` and `prediction`.

Step 4: Investigate anomalies

You can now filter and sort directly, without any re-joins:

```
where prediction = -1

sort anomaly_score desc

fields user, device_id, time_bucket, anomaly_score, prediction,
failure_rate
```

This produces results such as:

```
USER        | DEVICE_ID   | TIME_BUCKET | ANOMALY_SCORE |
PREDICTION | FAILURE_RATE

------------+-------------+-------------+---------------+--------
----+-------------

john.doe    | laptop-001  | 2026-01-29  | 87.3          | -
1         | 0.45

jane.smith  | mobile-042  | 2026-01-28  | 92.1          | -
1         | 0.67
```

You retain all the information needed to:

- Identify the affected users and devices.
- Understand when and where anomalies occurred.
- Take follow-up action (for example, pivot to raw events for the same user/device/time window).

Best Practices

Prefer explicit features

Where applicable, use explicit features for machine learning use cases, instead of using `fields` to drop context columns before apply.

```
... | apply my_model feature1 feature2 feature3
```

Keep identifiers and metadata in your aggregations

Wherever applicable, include user-related identifier fields in your aggregation clause, helping you preserve and leverage context for downstream analysis or scoring.

Use legacy behavior only when necessary

If you omit explicit feature names, `apply` will still attempt to use all columns as input to the model. This is only recommended if your pipeline exactly matches the schema used during training.

autoregress

Use the `autoregress` operator to prepare your events for autoregression or moving average calculations by duplicating one or more previous values for a field into each event.

Syntax

```
|autoregress <field> [as <newfield>] [auto_regress_
p={<int> | <int>-<int>}]
```

`<field>`	Required. Specify the column on which to perform the function.
`as <newfield>`	Optional. Specify a name of the new field. If you do not specify a name, the default name for the new field follows the form `<field>_p<num>`.
`auto_regress_p={<int> | <int>-<int>}`	Optional. Specify the number or number range of previous rows to be used as the basis for the interpolated values. If you specify a number (for example, `auto_regress_p=2`), the value from the second prior event log row is used. If you specify a number range (for example, `auto_regress_p=2-4`), then the values from the second, third, and fourth prior event log rows are used.

Usage

You can use the `autoregress` operator to prepare your events for autoregression or moving average calculations by duplicating one or more previous values for a field into each event. The initial events will not have prior value augmentation due to the absence of previous values.

> **Note:** Anomali recommends you use the "sort" on page 653 operator to order results before piping results to the `autoregress` operator.

Examples

Example 1: Duplicate previous row values for autoregression analysis.

```
|autoregress sales
```

Example 2: Add lagged values with a custom name and regression parameter.

```
|autoregress sales as agr auto_regress_p=4
```

Example 3: Include a range of prior row values for analysis.

```
|autoregress sales auto_regress_p=3-6
```

bin

Use the `bin` operator to specify aggregation windows.

Syntax

```
|bin <field> [as new_field] span = <integer>
[s,h,m,d,w,mon,y]
```

`span = [<integer> [s,h,m,d,w,mon,y]`	Multiple units are not allowed for week, month, or year. In other words, specify `1` for the integer place when specifying week, month, or year.

Usage

The `bin` operator is not necessary when you use the `timetrend` operator to perform aggregations over timespans. The `timetrend` operator has a `span` option that is similar to `bin`.

Examples

Example 1: Group events into 10-minute time windows.

```
|bin event_time span=10m
```

Example 2: Create 3-hour aggregation windows from timestamp field.

```
|bin timestamp span=3h
```

Example 3: Buckets OCSF events hourly and counts them by source port and user agent string, useful for spotting unusual port or user-agent combinations indicative of automated or malicious traffic.

```
ocsf | bin event_time span=1h | aggr count by event_time, src_
endpoint.port, http_request.user_agent
```

bucketdir

Use the `bucketdir` operator to group a field by its parent directory and return the number of instances of each parent directory.

Syntax

```
|bucketdir pathfield = <field> [dir_level =
<number>|<field>] [maxcount = <number>] [sep =
<string>]
```

`pathfield = <field>`	Required. Specify the field to group by parent directory.
`dir_level = <number>|<field>`	Optional. Specify an integer or field having an integer to set the number of directory levels to group by. If this option is not specified, the implicit default is `1`.
`maxcount = <number>`	Optional. Specify the number of rows to include in the summary. If this option is not specified, the implicit default is `NULL`.
`sep = <string>`	Optional. Specify the directory separator character. If this option is not specified, the implicit default is `/`.

Usage

You can use the `bucketdir` operator to view a summary of where a selection of logs are coming from and determine if some directories are producing less logging than expected or if some are producing more logging than expected.

Examples

Example 1: Group paths by their parent directory.

```
|bucketdir src
```

Example 2: Group by three levels of parent directories.

```
|bucketdir src dir_level = 3
```

Example 3: Group by dynamic directory depth calculation.

```
|bucketdir src dir_level = directory_depth - 1
```

Example 4: Limit summary to the top 100 directories.

```
|bucketdir src maxcount = 100
```

Example 5: Group by custom directory separator character.

```
|bucketdir pathfield=src sep=':'
```

calc

Use `calc` functions to create a new field and populate values for the new field with content returned by the function.

Syntax

```
|calc <new_field> = <function>
```

Usage and Examples

> **Note:** AQL boolean arguments are case-insensitive, that is, all the following boolean arguments are valid:
>
> **Positive case**: `True`, `true`, `t`

> **Negative case**: `False`, `false`, `f`

Type	Supported functions and syntax	Description

Comparison and Conditional functions	`case(<condition>,<value>, ...)`
	Example:
	`|where status!= ''`
	`|calc result = case(status = "401", "Unauthorized", status = "502", "bad gateway", status= "302", "url temperarliy moved")`
	`| fields status, result`
		Takes alternating conditions and values. Returns the first value for which the condition is true.

	`coalesce(<values>)`
	Example:
	`|where src != "" or src != "NULL" or src != "null" or src != "-"`
	`|calc res = coalesce(src, dest, "10.1.1.1")`
	`| fields src, dest, res`
		Takes one or more values. Returns the first non-null value.

	`if(<predicate>,<true_value>,<false_value>)`
	Example:
	`|calc result=if(status = '401', 'Unauthorized', 'not true')`
		If the `<predicate>` expression is true, returns the `<true_value>`. Otherwise returns the `<false_value>`.

	`insubnet (<cidr>,<ip>)`
	`<ip> insubnet (<cidr>)`
	`<ip> insubnet (<cidr1>,<cidr2>, ..., <cidrN>)`
	`<ip> insubnet (*)`
	Examples:
	Example 1:
	`|where dest_ip startswith "172.18"`
	`|fields src_ip, dest_ip`
	`|calc in_subnet=insubnet ("172.18.0.0/16", dest_ip)`
	Example 2:
	`ocsf | where dst_endpoint.ip startswith '10.' | fields src_endpoint.ip, dst_endpoint.ip | calc in_subnet=insubnet ('10.0.0.0/8', dst_endpoint.ip)`
		Returns TRUE if the specified CIDR address includes the value of the specified IP address field; otherwise, returns FALSE.
		Use the asterisk (*) symbol as an equivalent for '0.0.0.0/0' for matching any IP address.

Type	Supported functions and syntax	Description

	`like(<field>, <pattern>)`
	`not like(<field>, <pattern>)`
	Example:
	`|calc res = if(not like(user, 'test%'), 'found', 'not_found')`
		Evaluate whether the specified field matches the specified pattern. The pattern can specify an exact match or a wildcard match:
		• Use the percent (%) symbol as a wildcard for matching multiple characters.
		• Use the underscore (_) character as a wildcard to match a single character.

	`match(<field>, <regex>)`
	Example:
	`|calc does_field_match=if(match(user, '1{2,}'), "yes", "no")`
		Evaluate whether any substring of a specified field matches the specified regex.

Type	Supported functions and syntax	Description

	`null()`
	Example:
	`|calc null_action = null()`
	`|calc nulled_action=if(action = '', null(), action)`
		Use this function to set the value of fields with no value to `NULL`. You might want to do this when using aggregation functions to calculate the average or median of a field value.
		The `null()` function takes no arguments and returns `NULL`.
		When used with `if()`, as shown in the example, the `null()` function is processed if the condition is true. In the example, if `action` is blank, the `null()` function is processed and the `action` field is populated with `NULL`.

	`[subsearch]`
	Example:
	`| aggr count by sourcetype`
	`| appendtable [| where earliest=10d and latest=5d`
	`| aggr count by sourcetype]`
		Use this function to embed a search within the primary search, using square brackets (`[]`), to create more powerful queries that dynamically adapt according to the results of the embedded query.
		See Subsearch documentation for more information.

Type	Supported functions and syntax	Description

Text functions	`(<field or string> || <field or string>)`
	Example:
	`|calc res = ("Hello " || "there.")`
	returns a field name `res` having the phrase "Hello there."
	
	In the following example, assume the value for `field1` is test. The function returns a field name `res` having the phrase "This is a test."
	`|calc res = ("This" || " is a " || field1)`
		Concatenates the specified fields or strings into a new field. You must specify at least two arguments, and there is no upper limit. Order is important. The first argument is the start of the new field and subsequent arguments are appended in order.
		The `||` concatenation function is an alternative to the `strcat` function.

	`strcat(<field or string>, <field or string>[, ...])`
	Example:
	`|calc host_port = strcat(src, ":", src_port)`
	returns a field name `host_port` having values similar to "192.168.1.53:1813".
		Concatenates the specified fields or strings into a new field. Enclose string values in single or double quotes. You must specify at least two arguments, and there is no upper limit.
		The `strcat` function is an alternative to the `||` concatenation function.

Type	Supported functions and syntax	Description

	`len(<str>)`
	`length(<str>)`
	Example:
	`|fields dest, dest_ip, user |calc dest_length = len(dest)`
	`|fields url | calc url_length = length(url)`
		Returns the number of characters in the specified field value.
		The `<str>` argument can be the name of a string field or a string literal.

	`lower(<str>)`
	Example:
	`|calc res_lower=LOWER(sourcetype)`
		Converts values for the specified field to lowercase.

	`ltrim(<str>,<trim_chars>)`
	Example:
	`|where file_path = 'C:\Windows\System32\WindowsPowerShell\v1.0' |fields file_path | calc res_ltrim=LTRIM(file_path, 'C:\Windows')`
	`|calc x=ltrim(" ZZZZabcZZ ", " Z")`
		Removes characters from the left side of the specified field value.
		The `<str>` argument can be the name of a string field or a string literal.
		The `<trim_chars>` argument is optional. If not specified, spaces and tabs are removed from the left side of the string.

	`replace(<field>,<current_string>,<replacement_string>)`
	Example:
	`|where sourcetype = "Zscaler_leef" |fields sourcetype | calc res_replace=REPLACE(sourcetype, 'Zscaler_leef', 'Zleef')`
		Substitutes `<current_string>` with`<replacement_string>`for the specified field.

Type	Supported functions and syntax	Description

	`regex_replace(<field>,<regex>, <replacement_string>)`
	Example:
	`|where sourcetype="ms_defender" | calc test_regex_replace=regex_replace(src, "^qa-", "localTest-")`
		Replaces all occurrences of the `<regex>` string, using regular expression matches, with the replacement string. In this example, all occurrences of `^qa-` matches are replaced with `localTest-`.

	`rtrim(<str>,<trim_chars>)`
	Example:
	`|calc res_rtrim=RTRIM(sourcetype)`
		Removes characters from the right side of the specified field value.
		The `<str>` argument can be the name of a string field or a string literal.
		The `<trim_chars>` argument is optional. If not specified, spaces and tabs are removed from the right side of the string.

	`substr(<str>,<start>,<length>)`
	`substring(<str>,<start>,<length>)`
	Example:
	`|fields app_type |calc rest_sub=SUBSTR(app_type,5,25)`
		Returns a substring of the specified field value, beginning at the position specified by the `<start>` argument. The <length> argument specifies the number of characters to return.

Type	Supported functions and syntax	Description

	`trim(<str>,<trim_chars>)`
	Example:
	`|calc res_trim=TRIM(sourcetype)`
		Removes characters from both sides of the specified field value.
		The `<str>` argument can be the name of a string field or a string literal.
		The `<trim_chars>` argument is optional. If not specified, spaces and tabs are removed from both sides of the string.

	`upper(<str>)`
	Example:
	`|fields url |calc res_lower=UPPER(url)`
		Converts values for the specified field to uppercase.

	`urldecode(<str>)`
	Example:
	`|calc url_decoded=urldecode(url)`
	`|calc url_decoded=urldecode ('http%3A%2F%2Fi.just.love.urldecode')`
		Applies URL-decoding to convert specified URL fields or URL values to human-readable string values.
		For field values that are unencoded, the function returns the field name.

Type	Supported functions and syntax	Description

Conversion functions	`tonumber(<field>,<base>)`
	Example:
	`|calc ds_id_number = tonumber(ds_id)`
	`|calc ds_id_from_hex = tonumber(ds_id, 16)`
		Converts values for the specified string field to number values. The `<base>` option specifies the number base 2 to 36. The default if no base is specified is base 10.

	`tostring(<field>,<format>)`
	Example:
	`|calc dest_port_str = tostring(dest_port)`
		Converts values for the specified number or date field to a string value.

Type	Supported functions and syntax	Description

Date and Time functions	`from_unixtime(<bigint>)`
	Example:
	`|calc formatted_time = from_unixtime (event_time)`
		Converts a Unix timestamp into a human-readable date and time format.
		A Unix timestamp is a numeric representation of the number of milliseconds since January 1, 1970, 00:00:00 UTC.

	`to_unixtime(datetime_expression)`
	Example:
	`|calc unix_time = to_unixtime(from_ unixtime(event_time))`
		Converts a date and time value into a Unix timestamp.
		A Unix timestamp is a numeric representation of the number of milliseconds since January 1, 1970, 00:00:00 UTC.

Type	Supported functions and syntax	Description

	Note: `strftime()` and `timeformat()` are equivalent.
	`strftime(<time>, <format>)`
	`timeformat(<time>, <format>)`
	`|calc formated_time=strftime(field_str, '%Y-%m-%dT%H:%M:%S.%Q')`
	`|calc formated_time=timeformat(field_str, '%Y-%m-%dT%H:%M:%S.%Q')`
	`|calc formated_time=strftime(field_num, '%Y-%m-%dT%H:%M:%S.%Q')`
	`|calc formated_time=timeformat(field_num, '%Y-%m-%dT%H:%M:%S.%Q')`
	`|calc formated_time=strftime(field_num + 60 * 60 * 24, '%Y-%m-%dT%H:%M:%S.%Q')`
	`|calc formated_time=timeformat(field_num + 60 * 60 * 24, '%Y-%m-%dT%H:%M:%S.%Q')`
		Converts a Unix timestamp (in milliseconds) to a human-readable time in the specified format.
		See Date Time Format Code for more information.

	`strptime(<str>, <format>)`
	Examples:
	`|calc strp_time = strptime(event_time, "%Y-%m-%d %H:%M:%S")`
	`|calc strp_time = to_unixtime(strptime(event_time, "%Y-%m-%d %H:%M:%S"))`
		Converts a string that is a human-readable time to a Unix timestamp (in milliseconds). The string is parsed according to the format pattern you specify. See Date Time Format Code for more information.

Type	Supported functions and syntax	Description

	`relative_time(<time>,<expression>)`
	Example:
	`|calc relative_time_minus_1_day = relative_time(event_time, -1d)`
		Adjusts a Unix `<time>` value according to the specified relative time expression.
		A Unix timestamp is a numeric representation of the number of milliseconds since January 1, 1970, 00:00:00 UTC.

	`date_format(date_expression, format_string)`
	Example:
	`|calc formatted_time = date_format (event_time, '%Y-%m-%d')`
		Formats a date expression according to a specified format.
		See Date Time Format Code for more information.
		
		This function allows you to customize the format to be consistent with your organization's conventions. For example, you can use this function to change the order of day, month, and year components, including time, specifying the separator character, and more.

Type	Supported functions and syntax	Description

	`now()`
	Example:
	`|calc unit_time = now()`
		Returns the time that the search was started (in Unix time).
		A Unix timestamp is a numeric representation of the number of milliseconds since January 1, 1970, 00:00:00 UTC.

	`ctime()`
	Example:
	`|calc ctime_field = ctime(time_as_bigint)`
		Takes a bigint field as an argument and returns an epoch timestamp.

	`mktime()`
	Example:
	`|calc mktime_field = mktime(timestamp)`
		Takes a timestamp field as an argument and returns the epoch time.

Type	Supported functions and syntax	Description

Array functions	`array_contains(<field array>, <value>)`
	Example:
	`|calc n = array_contains(split(url, "/"), '1') | fields url,n`
		Returns true if `<value>` is present in the specified multivalue field or array.

	`split(<field>, <separator>)[num]`
	Example:
	`|calc url_res = split(url, '/')`
		Splits a field into an array.

	`regex_split(<field>,<regex>)`
	Example:
	`|where isnotnull(dest) | aggr count by dest | calc f1 = regex_split(dest, '\.') | where array_len(f1) > 1`
		Splits all occurrences of the regular expression matches in the field. In this example, all occurrences of the `'.'` symbol in the `dest` field will be separated.

	`array_len(array)`
	`array_count(array)`
	Example:
	`|calc n = array_len(split(url, "/")) | fields url,n`
	`|calc n = array_count(split(url, "/"))`
		Returns the length of an array.

	`array_concat(array1, array2)`
	Example:
	`|calc one_array=array_concat(dcid, ds_id)`
		Concatenates two arrays into one array, two strings into one string, or an array and a string (that appends the string to the array.)

	`array_distinct(array)`
	Example:
	`|calc distinct_emails = array_distinct (split(receiver, {","))`
		Removes duplicate values.

Type	Supported functions and syntax	Description

	`array_join(array, delimiter)`
	Example:
	`|calc emails = array_join(split (receiver, ","), ";")`
		Concatenates the individual values of the array using delimiter.

	`array_slice(array, start, length)`
	Example:
	`|where sender = "ck3@daisuke83.drkoop.site" or sender = "au@sralia.com" | calc first_batch = array_slice(split(sender,"@"), 1, 2)`
		Returns a subset from <field array> starting from index <start> with <length>.

Type	Supported functions and syntax	Description
JSON Extraction functions	**Note:** You must place `json_extract` after all possible condition clauses, to minimize the number of records processed. See AQL Best Practices for more information. `|where sourcetype = "ULink: aws_s3_dns" |calc host_name=json_extract(host, 'name')`	Creates a new field and populates values for the new field with content extracted from an existing field having JSON values. In the example: • `host_name` is the new field. • `json_extract` is the name of the function. • `host` specifies the event log field that contains the JSON. • `key` is the key part of the JSON key-value pair. Use single quotes to specify the JSON key.
	`json_extract(data, 'resource.instanceDetails.iamInstanceProfile.arn')`	Extract nested keys using dot notation to specify the path to the desired key.
	`ocsf | calc api_name=json_extract (actor.process.cmd_line, 'api_name')` **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.	Using the OCSF schema, extract JSON fields from process command lines. See "OCSF Schema Overview" on page 39 for more information.

Type	Supported functions and syntax	Description

	Examples:
	`json_extract(data, 'previous-year')`
	`json_extract(data, 'monitoring."start-date"')`
	`json_extract(data, 'log."number.alerts"')`
		If the JSON contains keys with special characters, the operator automatically skips the following characters:
		• `-` (dash)
		• `\` (backslash).
		If there are other special characters, avoid them by using double quotes (`" "`). Only the most nested key must be enclosed by double quotes. See examples.

	Examples:
	`json_extract(data, 'nonexistent')`
	`json_extract(data, 'key.nonexistent')`
		If the key does not exist in the JSON, `null` is returned.

Type	Supported functions and syntax	Description

	`json_extract(data, 'groups', ARRAY)`
	`json_extract(data, 'groups', BOOLEAN)`
	`json_extract(data, 'groups', NUMBER)`
	Examples:
	`|calc groups=json_extract(data, 'groups', ARRAY) |calc in_sg=array_contains(groups, 'sg-0abd9612be5dbaa33')`
	
	`|calc reason=json_extract(data, 'Log.labels.authorization', BOOLEAN)`
	
	`|calc reason=json_extract(data, 'Log.labels.authorization.value', NUMBER)`
		By default, the value extracted from the JSON is a string, even if the JSON value is an array. To retrieve results in a specific datatype, specify the datatype in the `json_extract` expression.
		For example:
		• To return an array, specify the ARRAY type in the function. This notation helps handle the extracted value as an array, which can be used with functions like `array_contains`.
		• To return a boolean status (true or false), specify the BOOLEAN type in the function.
		• To return a numeral, specify the NUMBER type in the function.

Type	Supported functions and syntax	Description

	Examples:
	`json_extract(data, 'resource.instance.interfaces{0}.ipAddress')`
	`json_extract(data, 'resource.instance.interfaces{}.ipAddress')`
	`json_extract(data, 'resource.instance.interfaces{}.ipAddress', ARRAY)`
		To extract nested keys from a specific element of an array, use the `{index}` notation. The index starts from 0.
		To extract all nested keys in the array of elements, the index can be omitted. This will extract the key for each element in the array. By default, this will return an array of IP addresses but as a string.
		In the example, the `ARRAY` type cFian be added to get all the IP addresses for the instance of the array type.

	Examples:
	`json_extract(object_extract, 'resource.instance.interfaces{}.groups{}.groupId')`
	`json_extract(object_extract, 'resource.instance.interfaces{}.groups{}.groupId', ARRAY)`
		This allows complex scenarios where keys of nested arrays need to be extracted.
		In the examples, the group IDs for all the groups for all the interfaces in the instance are extracted.

Type	Supported functions and syntax	Description

	Example of a JSON field with a nested JSON:
	`{"event": "{\"userIdentity\": {\"accountId\": \"3a99123\"}}"}`
	Full AQL:
	`|calc event = json_extract (message, 'event')`
	`|calc account_id = json_extract (event, 'userIdentity.accountId')`
		If the JSON field contains a nested JSON in string format, you must first extract the string key to be able to extract any key from the nested JSON.
		For example: if the `message` field contains a nested JSON called `event` in string format:
		1. Use `json_extract` and extract the event key.
		2. Use the `event` field toextract any nested key from the string.

Type	Supported functions and syntax	Description

	Note: Here are some important aspects of JSON filters to note:
	• Nested JSON filters are not supported.
	• Only = comparison is supported.
	• It returns results only if the key exists and is unique in the array. Otherwise, it returns `null`.
	
	Example of a JSON field with a nested JSON:
	`{"resource": {"privateIpAddresses": [{"privateDnsName":"ip-88-88-888-88.internal","ipAddress":"88.88.888.88"}]}}`
	
	Full AQL Example 1:
	`| calc ip=json_extract(message, "resource.privateIpAddresses{}(privateDnsName='ip-88-88-888-88.internal').ipAddress")`
	Here,
	• JSON Path to the array: `resource.privateIpAddresses{}`
	• Condition: `privateDnsName='ip-88-88-888-88.internal'`
	• JSON attribute to extract: `ipAddress`
		You can use a JSON filter if:
		1. There is a JSON array containing multiple objects
		2. One of those objects contains important information to extract
		3. The object position is unknown in the array.
		
		Here is the syntax for the JSON filter:
		1. Path to the JSON array.
		2. Condition enclosed in parenthesis, including the JSON key and value.
		3. (Optional) Path of the JSON key to extract. If the entire JSON object needs to be extracted, this can be omitted.
		
		Syntax:

Type	Supported functions and syntax	Description

	Result: `88.88.888.88`
	
	Full AQL Example 2:
	`| calc ip=json_extract(message, "resource.privateIpAddresses{}(privateDnsName='ip-88-88-888-88.internal')")`
	Here,
	• JSON Path to the array: `resource.privateIpAddresses{}`
	• Condition: `privateDnsName='ip-88-88-888-88.internal'`
	• JSON attribute to extract: No specific attribute to extract, therefore, the entire object is returned.
	Result:
	`{"privateDnsName":"ip-88-88-888-88.internal","ipAddress":"88.88.888.88"}`
		`<json-path>(<key>=<value>).<json-key>`

Type	Supported functions and syntax	Description

Hash functions	`md5(<field> | '<string>')`
	Example:
	`|calc string_input = md5('hello')`
		Convert string fields or string values into a hash using the MD5 cryptographic hashing algorithm.

	`sha1(<field> | '<string>')`
	Example:
	`|calc summary_raw = concat(src, src_port, dest, dest_port, sourcetype, aggr_col1_, totalbytes, earliest_time, latest_time, country, city, totalbytesMB)`
	`|calc d3_uuid = sha1(summary_raw)`
	`|fields src, src_port, dest, dest_port, sourcetype, aggr_col1_, totalbytes, earliest_time, latest_time, action, bytes_transferred_values, user, country, city, totalbytesMB, d3_uuid`
		Convert string fields or string values into a hash using the SHA1 cryptographic hashing algorithm.

	`sha256(<field> | '<string>')`
	Example:
	`|calc field_hash = sha256(sender)`
		Convert string fields or string values into a hash using the SHA256 cryptographic hashing algorithm.

	`sha512(<field> | '<string>')`
	Example:
	`|calc field_hash = sha512(sender)`
		Convert string fields or string values into a hash using the SHA512 cryptographic hashing algorithm.

Type	Supported functions and syntax	Description

Informational functions	`isnull(<field>, "yes", "no")`
	`isnotnull(<field>, "yes", "no")`
	Example:
	`|calc is_field_null=if(isnull(bytes_in), "yes", "no")`
	`|calc is_field_null=if(isnotnull(bytes_in), "yes", "no")`
		Use these functions to determine whether values for a specified field are NULL or not NULL.

	`isnum(<field>, "yes", "no")`
	Example:
	`|calc is_field_num=if(isnum(field), "yes", "no")`
		Use this function to determine whether the values for a specified field are numbers.

Type	Supported functions and syntax	Description

	`isoutlier(<field>, [param], [uselower])`
	Example:
	`|calc is_field_outlier=isoutlier(bytes_in, 1.5)`
	`|calc is_field_outlier=isoutlier(bytes_in, 1.5, true)`
		Use this function to determine whether value for a specified field is an outlier.
		Optional:
		• `param`: Specify a floating point value. This parameter multiplies the interquartile range (IQR) in the evaluation. Increasing the value yields fewer outliers; decreasing the value yields a larger number of outliers. If this parameter is not specified, the implicit default is 2.5.
		• `uselower`: Specify `true` to detect outliers below the median. If this parameter is not specified, the implicit default is `false`.

Type	Supported functions and syntax	Description

Mathematical functions	`abs(<number>)`
	Example:
	`|calc absnum=abs(-2)`
		Takes a number and returns its absolute value.

	`ceil(<field> | <number>)`
	`ceiling(<field> | <number>)`
	Example:
	`|calc upper_bound_val = ceil(decimal_value)`
	`|calc high_val_test = ceiling(2.1)`
		Converts numeric fields or numeric values to an upper bound numerical value that is the lowest integer above the given value. In other words, 2.1 is converted to 3.

	`floor(<field> | <number>)`
	Example:
	`|calc lower_bound_val = floor(decimal_value)`
	`|calc low_val_test = floor(2.1)`
		Converts numeric fields or numeric values to a lower bound numerical value that is the greatest integer below the given value. In other words, 2.1 is converted to 2.

	`power(<field> | <number)>`
	`pow(<field> | <number>)`
	Example:
	`|calc second_power = power(input_field, 2)`
	`|calc third_power = pow(input_field, 3)`
		Raises numeric fields or numeric values to the given power based on the second parameter of the function. In other words, power(2, 2) will produce 4, or power(2, 3) will produce 8.

Type	Supported functions and syntax	Description

	`round(<field> | <number> [, precision])`
	Example:
	`|calc round_to_int = round(decimal_value)`
	`|calc round_to_2_places = round(decimal_value, 2)`
	`|calc rounding_number = round(1.234)`
		Converts numeric fields or numeric values to a number having the decimal places specified by the given precision. If precision is not specified, it will round to the nearest integer.

	`|calc <new field> = sqrt(<field> | <number>)`
	Example:
	`|calc sqrt_numbers=sqrt(numbers)`
		Returns the square root of numeric fields or numeric values.

	`|calc <new field> = (<field> | <number> * <field> | <number>)`
	Example:
	`|calc myfield=(src_port * 80)|fields src_port,myfield`
		Returns the product of a numeric field or number multiplied by a numeric field or number.

	`|calc <new field> = (<field> | <number> / <field> | <number>)`
	Example:
	`|calc myfield=(src_port / 80)|fields src_port,myfield`
		Returns the product of a numeric field or number divided by a numeric field or number.

	`|calc <new field> = (<field> | <number> + <field> | <number>)`
	Example:
	`|calc myfield=(src_port + 80)|fields src_port,myfield`
		Returns the sum of a numeric field or number added to a numeric field or number.

	`|calc <new field> = (<field> | <number> - <field> | <number>)`
	Example:
	`|calc myfield=(src_port - 80)|fields src_port,myfield`
		Returns the difference of a numeric field or number subtracted from a numeric field or number.

Type	Supported functions and syntax	Description

Trigonometric functions	`<new field>=haversine(lat, lon, prev_lat, prev_lon, <distance_unit>)`
	Example:
	`|calc distance = haversine(tonumber(lat), tonumber(lon), tonumber(prev_lat), tonumber(prev_lon), 'mi')`
		Calculates the distance between sequential login events for a user, based on distances between latitude-longitude pairs within queries.
		Helps you determine impossible travel, that is, you can automatically identify geographically improbable logins compared to plausible travel speeds, indicating compromised logins.
		Supports string-based geolocation fields and integration with other AQL operators such as `iplocation` (converts IP addresses to geographical location lat-lon) and `streamstats` (combines coordinates on one line, with the previous coordinates for a user and the previous event time).
		Supported units of distance measurement: `"km"`

Type	Supported functions and syntax	Description
		(kilometers) and `"mi"` (miles).

Type	Supported functions and syntax	Description

Multivalue functions	`mvdedup(field)`
	Examples:
	`|calc a = mvdedup(ipaddresses)`
		Remove duplicate values in multivalue fields.

	`mvappend(<values>)`
	Examples:
	`|calc res=mvappend(src_ip, dest_ip)`
	`|calc res=mvappend(src_ip, dest_ip, 'localhost', '127.0.0.1')`
	`|calc res=mvappend('0.0.0.0', 'localhost', '127.0.0.1')`
	`|calc res=mvappend(['0.0.0.0', 'localhost', '127.0.0.1'], src_ip, dest_ip)`
	`|calc res=mvappend(1, 2, 3, 4)`
	`|calc res=mvappend([1, 2, 3, 4], 5, 6)`
	`|calc res=mvappend(1, array_number_field, 4)`
	`|calc res=mvappend('0.0.0.0', mvappend('localhost', '127.0.0.1'))`
		Returns a single multivalue result from a specified list of values.
		Specify any combination of string or number fields or values, including multivalue fields. Put multivalue arrays in brackets `[ ]`.
		You can also use nested `mvappend()` functions. The examples show required formatting.

	`mvcount(field)`
	Examples:
	`|fields event_time, url |makemv delim="/" url | calc total_count = mvcount(url_mv)`
		Returns a count of the number of elements in a list for the specified multivalue field.

	`mvfind(field, regex)`
	Examples:
	`|calc res=mvfind(ipaddresses, "^127.*$")`
	`|calc res=mvfind(commands, "login")`
	`|calc res=mvfind(ports, "443")`
	`|calc res = if(mvfind(ports, "8080") >= 0, 'found', 'not_found')`
	`|calc res=mvfind(ipaddresses, "^127.*$"), 'found', 'not_found'`
		Returns the index for the first value in a multivalue field that matches a regular expression. The index begins with zero. If no values match, returns `NULL`.

Type	Supported functions and syntax	Description

	`mvindex(<field>, <start>, <end>)`
	`array_element(<field>, <start>, <end>)`
	`array_elem(<field>, <start>, <end>)`
	Examples:
	`|calc res=mvindex(name_list_mv, 0, 2)`
		• `<field>` must be a multivalue field.
		• `<start>`: Index of the first value to include. The first value in a multivalue field is index 0, the second value index 1, and so on. -1 specifies "the last value in the list".
		• `<end>`: Index of the last value to include. -1 specifies "the last value in the list". `<end>` is optional.
		`mvindex()`, `array_element()`, and `array_elem()` are equivalent.

	`mvjoin(<field>, <delimiter>)`
	Examples:
	`|fields src |makemv delim = '.' src |calc ips = mvjoin(src_mv, '!!') |fields joined`
	`|fields src |makemv delim = '.' src |calc ips = mvjoin(src_mv, '-') |fields joined`
		Concatenates values within the multivalue field.
		• `<field>` must be a multivalue field.
		• `<delimiter>`: The string character in the multivalue field that separates the multiple values.

Type	Supported functions and syntax	Description

	`mvsort(field)`
	Example:
	`|calc res=mvsort(ipaddresses)`
		Sort the multivalue field lexicographically:
		• Numbers before letters.
		• Numbers are sorted by first digit. Example: 10, 8, 60, 100 are sorted lexicographically as 10, 100, 60, 8.
		• Uppercase before lowercase letters.

Type	Supported functions and syntax	Description

	`mvzip(<field1>, <field2>, <delim>)`
	Example:
	`|calc suspicios_nodes = mvzip (suspicious_ips_mv, hostnames_mv)`
	`|calc suspicios_users = mvzip (usernames, anomalous_activities, ' -> ') | fields suspicious_users`
		Creates a new multivalue field with concatenated values from two multivalue fields.
		• `<field1>` and `<field2>` must be multivalue fields.
		• `<delim>`: The string character in the multivalue field that separates the multiple values.
		The values in the first field are zipped with the values from the second field. If the number of values in the two fields does not match, the resulting multivalue has the length of the smaller list of values.

Type	Supported functions and syntax	Description

Information Theory functions	**Caution:** Do not use `shannon_entropy` directly on the raw `message` field. Doing so can cause excessive resource consumption and produce less meaningful results. Use specific fields, such as `url`, `filename`, or `process`, for better performance and more actionable scores.
	
	`... | calc <entropy_score> = shannon_entropy (<string_to_evaluate>)`
	
	Examples:
	`eventlog | limit 1000 | calc entropy = shannon_entropy(url)`
	`eventlog | limit 1000 | calc entropy = shannon_entropy(url) | where entropy > 7.5`
	`eventlog | calc column_1 = shannon_entropy(sourcetype) | calc column_2 = shannon_entropy(user)`
	
	Note: Each input row is capped at 50,000 characters per `shannon_entropy` call. Values that exceed this limit are truncated before the entropy score is calculated.
		Calculates an entropy score for a string field. The score measures the randomness or unpredictability of the characters in the field value.
		
		Parameters
		
		• `entropy_score`: The name of the new column that stores the entropy score. The score is a decimal value between zero and eight.
		
		• `string_to_evaluate`: An existing column with a string type to evaluate. Use specific fields such as `url`, `filename`, or `process` for best results. There are no option parameters or default values for this operator.
		
		Entropy Score
		
		1. The function scans each character in the input string.

Type	Supported functions and syntax	Description
	Note: If the input column contains an empty or NULL value for a row, `shannon_entropy` returns a score of zero for that row.	2. It calculates how often each character appears relative to the total length of the string. 3. It assigns a score based on how evenly distributed, and therefore unpredictable, the characters are. Strings with many different characters appearing in roughly equal proportions produce a high score. Strings that repeat the same characters frequently produce a low score. The score is a decimal value from zero to eight: • **High Entropy (7.5–8.0):** Likely encrypted, compressed, or packed files. • **Low Entropy (below 5.0):** Likely plain text or structured data.

Type	Supported functions and syntax	Description
		Use cases • **Malware Detection**: Packed or encrypted malware has high entropy (`entropy score > 7.5`). • **Data Exfiltration**: Detecting encrypted archives in network traffic. • **Ransomware Detection**: Encrypted files have uniform high entropy. • **SIEM Integration**: Entropy is useful as a detection feature for file activity logs.

UUID function	`<new field>=uuid()`
	Example:
	`|calc id=uuid()`
		Generates a new column with UUIDs.

chart

Use the `chart` operator to transform the results of a search into table data that can be readily displayed as a one of the AQL dashboard panel visualizations.

Syntax

```
|chart <aggregation expression> by <field1> [,
<field2>] [limit=<int>]
```

`<aggregation expression>`	Required. Specify an aggregation expression to be added to the results table. Only one aggregation is supported. See "aggr" on page 436 for information about aggregation functions.
`<field1>`	Required. Specify the field to group data by rows.
`<field2>`	Optional. If specified, then the resulting data is pivoted by creating a cross table: • The values in `<field1>` will be located in the rows. • The values in `<field2>` will be located in the columns.
`limit=<int>`	Optional. Specify how many columns will be displayed in the results table. If there are more values than the specified limit, then the results that exceed the limit will be grouped in an additional column called `others`. Default: `10`.

Usage

You can use the `chart` operator to apply aggregations by different fields, transforming the data into a cross table (row/columns) if needed.

The `chart` operator supports the same aggregation functions as the "aggr" on page 436 operator (for example, sum, count, avg, and so on).

Examples

Example 1: Count events grouped by application type.

```
|chart count BY app_type
```

Example 2: Sum event sizes for each application type.

```
|chart sum(event_size) BY app_type
```

Example 3: Create a cross-tabulation of event sizes by type and volume.

```
|chart sum(event_size) BY ap
```

Example 4: Produces a cross-tabulation chart of event counts by product name and activity type, useful for comparing activity distribution across security sources at a glance.

```
ocsf | chart count by metadata.product.name, activity_name
```

cluster

Use the `cluster` operator to generate clusters of data based on similarity.

Syntax

```
|cluster [threshold=<float> | match=<string> |
delim=<string> | countfield=<string> |
labelfield=<string> | field=<string>]
```

`threshold=<float>`	Optional. Specify a numeric value between 0 and 1 to define the sensitivity of the clustering. If this value is closer to 1, then it will group only more similar data. Otherwise, it will group less similar data getting fewer clusters.

`match=<string>`	Optional. Specify the match type to be used for the clustering. Possible options: • `termlist`: Uses the delimiters option to compare the list of terms by position. • `termset`: Uses the delimiters option to compare all the distinct terms ignoring the order of them. • `jaccard`: Uses the Jaccard comparison to find the similarity among the data. • `jaro`: Uses the Jaro comparison to find the similarity among the data. Default: `termlist`
`delim=<string>`	Optional. Specify the delimiters to be used for `termlist` and `termset` match types. Default: `^A-Za-z0-9_`
`countfield=<string>`	Optional. Specify the name of the output field for the number of events in the cluster. Default: `cluster_count`
`labelfield=<string>`	Optional. Specify the name of the output field for the cluster name. Default: `cluster_label`
`field=<string>`	Optional. Specify the name of the field from which the data will be compared. Default: `message`

Usage

You can use the `cluster` operator for data onboarding.

Examples

Example 1: Cluster data using default similarity matching.

```
test_cluster_table |cluster
```

Example 2: Cluster with custom matching algorithm and threshold settings.

```
test_cluster_table |cluster countfield=counts labelfield=label_
field threshold=0.6 field=message match=jaccard delim=',-:'
```

Example 3: Generate clusters using term-set comparison method.

```
test_cluster_table |cluster match=termset
```

Example 4: Clusters the first 500 OCSF events by `metadata.product.name` at 60% similarity, storing cluster labels in `nb_cluster` and the count of events per cluster in `test_count`.

```
ocsf | limit 500 | cluster labelfield=nb_cluster countfield=test_
count threshold=0.6 field=metadata.product.name
```

cofilter

Use the `cofilter` operator to find the relation between the values in two fields.

Syntax

`|cofilter <field1> <field2>`

`<field1>`	Specify the field to find the popularity of its values in `<field2>`. The values of this field will be analyzed and the occurrences will be compared among the values of `<field2>`.

`<field2>`	Specify the field to be included in the resulting comparison. Basically, all its values are compared among them by comparing the occurrences of values of `<field1>`.

Usage

You can use the `cofilter` operator to to find the popularity of values in `<field1>` and `<field2>`.

The `cofilter` operator returns the following fields in results:

`<field2>1`	The name of the `<field2>` with a 1 as a suffix. It represents the first value of `<field2>` to compare against the second value of `<field2>`.
`<field2>1_count`	The number of different occurrences of `<field1>` for the value in `<field2>`.
`<field2>2`	The name of the `<field2>` with a 2 as a suffix. It represents the second value of `<field2>` that is being compared.
`<field2>2_count`	The number of different occurrences of `<field1>` for the value in `<field2>`.
`pair_count`	The number of different values for `<field1>` that are being compared.

Examples

Example 1: Compare user popularity across destination addresses.

```
|cofilter user dest
```

Example 2: Analyze correlation between users and URL access patterns.

```
|cofilter user url
```

Example 3: Find relationship between field values through co-occurrence.

```
|cofilter src_ip user
```

concurrency

Use the `concurrency` to count the number of events with the spans that overlap with the start of each event.

Syntax

```
table_name|concurrency concurrency_duration=<field>
[concurrency_start=<field>][concurrency_
output=<field>]
```

`concurrency_duration=<field>`	Required. Specify the numeric field that contains the duration taken by an event. The units in this field must match the units in the field used as `concurrency_start`.
`concurrency_start=<field>`	Optional. Specify the field that denotes the event start. Default: `event_time`
`concurrency_output=<field>`	Optional. Specify the field in which the concurrency calculation results will be stored. Default: `concurrency`

Usage

You can use the `concurrency` to calculate the count of events going on at the moment when each event starts, including the event itself. This calculation does not provide the overall number of events with which a specific event overlaps throughout its entire duration.

Examples

Example 1: Calculate concurrent event overlap using duration field.

```
table_name|concurrency concurrency_duration=duration
```

Example 2: Count concurrent events from a custom date field.

```
table_name|concurrency concurrency_duration=duration concurrency_
start=event_dates
```

Example 3: Store concurrency results in a custom output field.

```
table_name|concurrency concurrency_duration=duration concurrency_
start=event_dates concurrency_output=con
```

Example 4: Calculates the number of concurrently active events at each point in time across the first 50 OCSF events, using `event_time` as both the start and duration reference.

```
ocsf | limit 50 | concurrency concurrency_duration=event_time
concurrency_start=event_time
```

contingency (alias: ctable)

Use the `contingency` operator to generate a cross table to find the number of occurrences for the values of two fields.

The `contingency` operator has an alias— `ctable`. You can use `contingency` and `ctable` operators interchangeably.

Syntax

```
|contingency <field1> <field2> [maxresults=<int> |
addtotals=<bool>]
```

```
|ctable <field1> <field2> [maxresults=<int> |
addtotals=<bool>]
```

`<field1>`	Required. Specify the first field to be used in the results table as rows.
`<field2>`	Required. Specify the second field to be used in the results table as columns.

`maxresults=<int>`	Optional. Specify an integer value to limit the number of rows returned in the results table. Default: `1000`
`addtotals=<bool>`	Optional. Specify a boolean value to add totals (rows and columns) to the results table. Default: `True`

Usage

You can use the `contingency` operator to find the relationship between the values of two fields by finding the number of occurrences (count aggregation) when the values appear.

Examples

Example 1: Create cross-tabulation of source and destination IP addresses.

```
|contingency src_ip dest_ip
```

Example 2: Generate contingency table without totals and limited results.

```
|contingency src_ip dest_ip addtotals=false maxresults=100
```

Example 3: Filters OCSF events to those with a non-empty source IP, then builds a contingency table showing how often each product generated traffic from each source IP.

```
ocsf | where src_endpoint.ip !='' | fields src_endpoint.ip,
metadata.product.name |contingency metadata.product.name src_
endpoint.ip
```

convert

Use the `convert` operator to rotate the rows and column values.

Syntax

```
|convert int=<int> columnname=<string> includeempty=
{boolean} headerfield=<field>
```

`int=<int>`	Number of rows in the original table to use as columns in the new results table. Default: `5` Max: 50
`columnname=<string>`	The name of the first column, containing the column names of the table that was converted. Default: `'column'`
`includeempty={true|false}`	Skip columns with empty values. Default: `true`
`headerfield=<field>`	Specify the field to be used as header.

Usage

You can use the `convert` operator to rotate a table such that the rows and columns are swapped. Returns the column names as a column, `<columnname>`, and the rows as columns named `row1`, `row2` , and so on.

Examples

Example 1: Transpose table with default settings.

```
|convert
```

Example 2: Rotate table with six rows as columns.

```
|convert int=6
```

Example 3: Transpose with custom column naming and empty field inclusion.

```
|convert int=3 columnname='test' includeempty=true
```

Example 4: Convert with custom header field from sourcetype.

```
|convert int=6 columnname='columns' headerfield=sourcetype
```

Example 5: Transpose with extended row count as columns.

```
|convert int=10
```

creinv

Use the `creinv` operator to create a new ThreatStream investigation.

Syntax

```
|creinv inv_name='<string>' assignee=<number>
assignee_type='{user|tsworkgroup}' [ts_
users=<numbers>] [workgroups=<numbers>] [inv_
description='text'] [fields=<fields>]
```

`inv_name='<string>'`	Specify the investigation name.
`assignee=<number>`	Specify the assignee ID for the investigation. It can be a user id or a workgroup id.
`assignee_type='{user|tsworkgroup}'`	Specify whether the assignee is a `user` or a `tsworkgroup`.
`[ts_users=<numbers]`	Optional. Specify a comma-separated list of user IDs to grant access to the investigation.
`[workgroups=numbers]`	Optional. Specify a comma-separated list of workgroup IDs to grant access to the investigation.
`[inv_description='text']`	Optional. Specify investigation description text.
`[fields=<fields>]`	Optional. Specify the names of result fields sent as candidate observables.

Usage

You can use the `creinv` operator to create a new ThreatStream investigation. Use the `listinv` operator to retrieve a list of ThreatStream investigations. Use the `addinv` operator to add elements to an existing ThreatStream investigation.

For information about ThreatStream investigations, refer to ThreatStream online help.

Examples

Example 1: Create a new investigation with user and workgroup access.

```
|creinv inv_name='test_inv' assignee=1 assignee_type='user' ts_
users=1,2,3 workgroups=4,5 fields=int_col,str_col inv_
```

Example 2: Takes a single OCSF event, selects key forensic fields across activity, endpoint, user, and process context, then creates an investigation named `test_inv` assigned to user ID = `1`.

```
ocsf | limit 1 | fields activity_name, metadata.product.name,
src_endpoint.hostname, src_endpoint.port, actor.user.name,
actor.process.cmd_line, actor.process.file.name,
actor.process.parent_process.file.name, device.os.type  | creinv
inv_name='test_inv' assignee=1 assignee_type='user'
```

CrowdStrike Operators

> **Note:** Access to CrowdStrike operators is provided on a limited-availability basis. Contact Anomali Customer Support for more information.

Anomali supports integration with CrowdStrike services, enabling you to access CrowdStrike data through Anomali Search. Refer to Integration with CrowdStrike for instructions on activating the integration.

After the integration is activated, you can use the following operators to query CrowdStrike data in Anomali Search:

- cs_query_alerts

> **Note:** The `cs_query_alerts` operator replaces the `cs_query_detects` operator, which is now deprecated. See "Migration" on page 512 for migration details.

- cs_get_alert_entities

> **Note:** The `cs_get_alert_entities` operator replaces the `cs_get_detect_summaries` operator, which is not deprecated. See "Migration" on page 514 for migration details.

- cs_query_device
- cs_get_system_info
- cs_get_process_detail
- cs_hunt_ip
- cs_hunt_domain
- cs_hunt_file
- cs_list_users

cs_query_alerts

> **Note:**
>
> - Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.
> - The `cs_query_alerts` operator replaces the `cs_query_detects` operator, which is now deprecated. See "Migration" on page 512 for details.

Use the `cs_query_alerts` operator to search for alerts using a filter specified in CrowdStrike Falcon Query Language (FQL).

The search results are returned in the `resources` array (displayed as the RESOURCES column in the UI). You can reference this array in subsequent CrowdStrike operators, such as `cs_get_alert_entities`, to retrieve alert details for the listed alerts.

Syntax

```
| cs_query_alerts [cs_filter=<filter> cs_sort=<sort_criteria> cs_limit=<max_results> cs_offset=<offset> cs_include_hidden=<boolean>]
```

`cs_filter=<filter>`	Optional. Specify a CrowdStrike Falcon Query Language (FQL) filter expression to reduce the number of results. The expression format is as follows: `cs_filter="<filter>:<filter_value>"` For example: • `status:'new'` • `severity:>=50`
`cs_sort=<sort_criteria>`	Optional. Specify a sort criteria by which to sort the results. Possible values, and the sort order: • `first_behavior`: Timestamp of the first behavior associated with this detection. • `last_behavior`: Timestamp of the last behavior associated with this detection. • `max_severity`: Highest severity of the behaviors associated with this detection. • `max_confidence`: Highest confidence of the behaviors associated with this detection. • `adversary_id`: ID of the adversary, if any, associated with this detection. • `devices.hostname`: Hostname of the host where this detection was detected. Specify the sort order: `asc` | `desc`

`cs_limit=<max_results>`	Optional. Specify the maximum number of results to return. Default value: `100` Maximum value: `10,000`
`cs_offset=<offset>`	Optional. Specify the number of results to skip before returning results. Use in conjunction with `cs_limit=<max_results_return>` to paginate results. Refer to "Offsetting Query Results" below for details. Default value: `0`
`cs_include_hidden=<boolean>`	Optional. Specify whether to include hidden alerts. Possible values: • `true` • `false` Default value: `true`

Offsetting Query Results

When using pagination with the `cs_query_alerts` operator, the combined value of `cs_limit` and `cs_offset` must not exceed 10,000. This restriction is enforced by the CrowdStrike API and defines the maximum accessible result window. As a result, pagination is supported only within the first 10,000 matching records.

If a query returns more than 10,000 results, additional records cannot be retrieved using offset-based pagination alone. To access the full dataset, refine the query using filters (for example, time range, severity, or other attributes) to reduce the result set and page through each filtered subset separately.

Examples

Example 1: Query new alerts with a limit of 500 results.

```
| cs_query_alerts cs_filter="status:'new'" cs_limit=500
```

Example 2: Query high-severity alerts (≥ 50) sorted by last behavior in descending order.

```
| cs_query_alerts cs_filter="severity:>=50" cs_sort="last_
behavior|desc"
```

Example 3: Query up to 300 new alerts and skip the first 100 matching alerts.

```
| cs_query_alerts cs_filter="status:'new'" cs_limit=300 cs_
offset=100
```

Example 4: Query five alerts, retrieve output in `resources`, extract process IDs that triggered the alerts, and get details about these process IDs in `processid`. For more information on `resources`, see the operator description above.

```
| cs_query_alerts cs_limit=5

| cs_get_alert_entities cs_id_list=resources

| calc processid = json_extract(response, "triggering_process_
graph_id")

| cs_get_process_detail cs_id_list=processid
```

Migration

The `cs_query_detects` operator is deprecated and replaced by `cs_query_alerts`. If you currently use `cs_query_detects` to retrieve alert data, update your queries to use `cs_query_alerts`.

Use the following example to migrate existing queries to the `cs_query_alerts` operator:

Before	After
`| cs_query_detects cs_filter="status:'new'"`	`| cs_query_alerts cs_filter="status:'new'"`

cs_get_alert_entities

> **Note:**
>
> - Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.
> - The `cs_get_alert_entities` operator replaces the `cs_get_detect_summaries` operator, which is now deprecated. See "Migration" on the next page for details.

Use the `cs_get_alert_entities` operator to retrieve full alert details by a composite ID.

Syntax

```
| cs_get_alert_entities cs_id_list=<composite_ids>_or_<field_reference> [cs_include_hidden=<boolean>]
```

`cs_id_list=<composite_ids>_or_<field_reference>`	Specify composite IDs or a field reference. `<composite_ids>` is either a single composite ID or a comma-separated list of composite IDs, without spaces, and enclosed in double quotes. `<field_reference>` is the `resources` array with composite IDs returned in the output of the `cs_query_alerts` operator. See "cs_query_alerts " on page 509 for more information.
`cs_include_hidden=<boolean>`	Optional. Specify whether to include hidden alerts. Possible values: • `true` • `false` Default value: `true`

Examples

Example 1: Retrieve alert details using a specific composite ID.

```
| cs_get_alert_entities cs_id_list="ldt:abc123:1001"
```

Example 2: Retrieve alert details of two composite IDs.

```
| cs_get_alert_entities cs_id_
list="ldt:xyz345:1002,ldt:ntp584:3456"
```

Example 3: Query and filter new alerts, store them in `resources`, and pipe the results to retrieve their full details. For more information on resources, see cs_query_alerts.

```
| cs_query_alerts cs_filter="status:'new'" | cs_get_alert_
entities cs_id_list=resources
```

Migration

The `cs_get_detect_summaries` operator is deprecated and replaced by `cs_get_alert_entities`.

If you currently use the `cs_get_detect_summaries` operator to retrieve full alert details, use the following example to correctly migrate your existing queries to the `cs_get_alert_entities` operator:

Before	After
`| cs_get_detect_summaries cs_id_list="id1,id2"`	`| cs_get_alert_entities cs_id_list="id1,id2"`

cs_query_device

Note: Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.

Use the `cs_query_device` operator to query devices in your environment. You can search by platform, hostname, IP address, and other attributes.

The device identifiers are returned in the AID output field. You can use this field with the `cs_hunt_domain`, `cs_hunt_file`, `cs_hunt_ip`, and `cs_query_device` operators to perform analysis and threat hunting.

Syntax

```
| cs_query_device [cs_filter=<filter> cs_limit=<max_results>]
```

`cs_filter=<filter>`	Optional. Specify a CrowdStrike Falcon Query Language (FQL) filter expression to reduce the number of results. The expression format is as follows: `cs_filter="<filter>:<filter_value>"` For example: • `status:'new'` • `severity:>=50`
`cs_limit=<max_results>`	Optional. Specify the maximum number of device AIDs to return. Default value: `5000` Maximum value: `10,000`

Examples

Example 1: Retrieve the device with the specified host name.

```
| cs_query_device cs_filter="hostname:'WS01'"
```

Example 2: Retrieve devices with the specified platform name and get system details for them.

```
| cs_query_device cs_filter="platform_name:'Windows'"
| cs_get_system_info cs_id_list=AID
```

cs_get_system_info

> **Note:** Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support fo more information.

Use the `cs_get_system_info` operator to view details of a specific device based on the device ID.

Syntax

```
| cs_get_system_info cs_id_list=<device_ids>
```

`cs_id_list=<device_ids>`	Specify a device ID or a list of device IDs separated by commas, without spaces, and enclosed in double quotes.

Examples

Example 1: View details of the specified device.

```
| cs_get_system_info cs_id_
list="1234dde1234b1d23ababb1b2b123c2a3"
```

Example 2: View details of the specified devices.

```
| cs_get_system_info cs_id_
list="1234dde1234b1d23ababb1b2b123c2a3,1234dde1234b1d23ababb1b2b1
23c2a4"
```

cs_get_process_detail

> **Note:** Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.

Use the `cs_get_process_detail` operator to retrieve process details of specified process IDs.

Syntax

```
| cs_get_process_detail cs_id_list=<process_ids>
```

`cs_id_list=<process_ids>`	Specify a process ID or a list of process IDs separated by commas, without spaces, and enclosed in double quotes. The format of a process ID is the following: `pid:<aid>:<pid>` Maximum number of process IDs that can be specified for a search operation is 100. If more than 100 process IDs need to be processed, run multiple searches.

Examples

Example 1: Retrieve process details of the specified process ID.

```
| cs_get_process_detail cs_id_
list="pid:1234dde1234b4d60ababb3b8b942c2a3:1234561234567"
```

Example 2: Retrieve process details of the specified process IDs.

```
| cs_get_process_detail cs_id_
list="pid:aaa:111,pid:bbb:222,pid:ccc:333"
```

cs_hunt_ip

> **Note:** Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.

Use the `cs_hunt_ip` operator to get a list of device AIDs on which the IP was matched.

The device identifiers are returned in the AID output field. You can use this field with the `cs_hunt_domain`, `cs_hunt_file`, `cs_hunt_ip`, and `cs_query_device` operators to perform analysis and threat hunting.

Syntax

```
| cs_hunt_ip cs_value=<ip_address> [cs_type=<ip_type> cs_
limit=<max_results>]
```

`cs_value=<ip_address>`	Specify an IP address enclosed in double quotes.
`cs_type=<ip_type>`	Optional. Specify an IP type. Possible values: • `ipv4` • `ipv6` Default: `ipv4`
`cs_limit=<max_results>`	Optional. Specify the maximum number of results to return. Default value: `100` Maximum value: `10,000`

Examples

Example 1: Retrieve a list of device AIDs on which the specified IP was matched.

```
| cs_hunt_ip cs_value="8.222.196.43"
```

Example 2: Retrieve a list of device AIDs on which the specified IP of the IPv4 type was matched. Limit the returned results to 2000.

```
| cs_hunt_ip cs_value="10.0.0.1" cs_type="ipv4" cs_limit=2000
```

Example 3: Retrieve device AIDs on which the specified IP was matched and then get their device details.

```
| cs_hunt_ip cs_value="8.8.8.8" cs_limit=500
| cs_get_system_info cs_id_list=AID
```

cs_hunt_domain

> **Note:** Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.

Use the `cs_hunt_domain` to get a list of device AIDs on which the specified domain was matched.

The device identifiers are returned in the AID output field. You can use this field with the `cs_hunt_domain`, `cs_hunt_file`, `cs_hunt_ip`, and `cs_query_device` operators to perform analysis and threat hunting.

Syntax

```
| cs_hunt_domain cs_domain_name=<domain_name> [cs_limit=<max_results>]
```

`cs_domain_name=<domain_name>`	Specify a domain name enclosed in double quotes.
`cs_limit=<max_results>`	Optional. Specify the maximum number of results to return. Default value: `100` Maximum value: `10,000`

Examples

Example 1: Retrieve the list of device AIDs matched on the specified domain.

```
| cs_hunt_domain cs_domain_name="google.com"
```

Example 2: Retrieve the list of device AIDs matched on the specified domain. Limit the returned results to 500.

```
| cs_hunt_domain cs_domain_name="malicious-domain.com" cs_limit=500
```

Example 3: Retrieve device AIDs on which the specified domain was matched and then get their device details.

```
| cs_hunt_domain cs_domain_name="google.com" | cs_get_system_info
cs_id_list=AID
```

cs_hunt_file

> **Note:** Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.

Use the `cs_hunt_file` operator to get a list of device AIDs on which the specified file hash was matched.

The device identifiers are returned in the AID output field. You can use this field with the `cs_hunt_domain`, `cs_hunt_file`, `cs_hunt_ip`, and `cs_query_device` operators to perform analysis and threat hunting.

Syntax

```
| cs_hunt_file cs_value=<hash_value> [cs_type=<hash_type> cs_
limit=<max_results>]
```

`cs_value=<hash_value>`	Specify a hash value enclosed in double quotes.
`cs_type=<hash_type>`	Optional. Specify a hash type. Possible values: • `sha256` • `md5` Default value: `md5`
`cs_limit=<max_results>`	Optional. Specify the maximum number of results to return. Default value: `100` Maximum value: `10,000`

Examples

Example 1: Retrieve the list of device AIDs matched on the specified file hash.

```
| cs_hunt_file cs_value="ae87a276ea911dbabbf91521bf72c943"
```

Example 2: Retrieve the list of device AIDs matched on the file hash with the specified value and type. Limit the returned results to 200.

```
| cs_hunt_file cs_
value="e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b78
52b855" cs_type="sha256" cs_limit=200
```

Example 3: Retrieve device AIDs on which the specified file hash file was matched and then get their device details.

```
| cs_hunt_file cs_value="99934t09g09i09d0dfghkjhkf88893000gjs7"

| cs_get_system_info cs_id_list=AID
```

cs_list_users

> **Note:** Access to this operator is provided on a limited-availability basis. Contact Anomali Customer Support for more information.

Use the `cs_list_users` operator to retrieve the list of users' UUIDs associated with your CrowdStrike Customer ID.

Syntax

```
| cs_list_users
```

Usage

The `cs_list_users` operator can be used to create a lookup table containing the list of users' UUIDs associated with the customer ID.

To create a lookup table with the list of users, export the search results to a CSV file and then create a lookup table using this CSV file.

For details on how to export the search results, see "Exporting Search Results" on page 86.

For details on how to create a lookup table from a CSV file, see "Creating Custom Lookup Tables " on page 136.

Example

```
| cs_list_users
```

curl

Use the `curl` operator to post URL content to a specified URI, just like the Linux curl command.

Syntax

`|curl [options...] <url>`

`-X` or `-request`	Request method: GET, POST, PUT, DELETE, HEAD, OPTIONS, PATCH.
`-H` or `-header`	Add header. Format: `Header-Name: Value`. Can be used multiple times.
`-d` or `-data`	Specify data.
`include`	Include headers in output.
`head`	Return only headers.
`user`	Define user:password.
`insecure`	Disable certificate verification.
`<url>`	URI

Usage

You can use the `curl` operator to post URL content to a specified URI, just like the Linux curl command.

In most cases, the piped `curl` operator is used to publish refined search results to the specified URI. In particular, you want to use the preceding AQL syntax to define

the result columns that you want to send. The `curl` operator posts the results in json format:

```
'{column_name: ["data", "data", ...]}'
```

Examples

Example 1: Post JSON data to a specified URI using HTTP.

```
|curl -X POST -include -d '{"key": "
```

dedup

> Do not apply the `dedup` operator to the `message` field. Doing so impacts search performance because each raw event is held in memory for this operation. Anomali recommends following the "AQL Best Practices" on page 409 to reduce the dataset size by limiting or filtering it before using `dedup`.

Use the `dedup` operator to remove duplicate records or entries from a dataset and leave only unique entries based on specified criteria.

Syntax

```
|fields <field> [,<field>,...] |dedup <field>
[,<field>, ...]
```

Usage

You must place the `fields` operator prior to the `dedup` operator in your query to ensure accurate functionality.

Example

Example 1: This shows one event log for each sourcetype.

```
|fields sourcetype  |dedup sourcetype
```

Example 2: This shows one event log for each action.

```
|where action is not null |fields action |dedup action
```

Example 3: Filter privileged user accounts and replace specific username values.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

```
ocsf | where actor.user.name IN ('root', 'Administrator',
'system') | dedup actor.user.name | replace 'system' with 'test_
of_system_replace' in actor.user.name
```

deletemodel

Use the `deletemodel` operator to delete a model that was generated by the `fit`operator.

Syntax

`|deletemodel '<model_name>'`

`<model_name>`	Name of the model to be deleted. Model name is case sensitive.

Usage

You can use the `deletemodel` operator to delete models that are no longer used.

Examples

Example 1: Delete a trained machine learning model.

```
|deletemodel 'model1'
```

domaintools

Use the `domaintools` operator to enrich a domain value or field containing domain values using the ThreatStream DomainTools Iris Investigate enrichment.

Syntax

`|domaintools <field or value>`

`<field or value>`	Specify the field name or domain to look up.

Usage

You can use the `domaintools` operator to enrich a number of domains and manage the observables based on the data returned from the enrichment. For example, you might want to tag domains with an Overall Risk Score greater than 80 as “dt_high_risk” and import them into ThreatStream as observables that are ingested by Integrator and sent to other security products as required.

Examples

Example 1: Retrieve domain intelligence data.

```
|domaintools url
```

Example 2: Enrich domains with third-party intelligence.

```
|domaintools "www.google.com"
```

erex

Use the `erex` operator to extract data based on a regular expression built from a list of example values that you specify.

Syntax

```
|erex <string> in=<string> [not=<string>]
[from=<string>] [output=<string>]
```

`<string>`	Specify the name of the new field where the extracted data will be written.
`in=<string>`	Specify a comma-separated list of example values to build the regular expression.
`[not=<string>]`	Optional. Specify a comma-separated list of example values to avoid when building the regular expression.
`[from=<string>]`	Optional. Specify the field from which the data will be extracted. If you do not specify this option, the implicit default is the raw message field `message`.
`[output=<string>]`	Optional. Specify this option to include a column in results that shows the regular expression that was built by the operator and used to extract data for the new field.

Usage

You can use the `erex` operator when there is a field having too much data, and you are only interested in part of it. Instead of manually extracting the data, you can extract it using this operator to automatically build a regular expression that extracts the data you want.

Examples

Example 1: Extract regex matches from external URLs.

```
|erex website
in='https://www.google.com,https://www.yahoo.com,https://www.semi
complete.com' |fields id, website
```

Example 2: Parse remote content with regex patterns.

```
|erex ip in='82.247.137.135,108.32.74.68,92.239.20.53' not='1.1'
|sort ip desc
```

Example 3: Extract structured data from web sources.

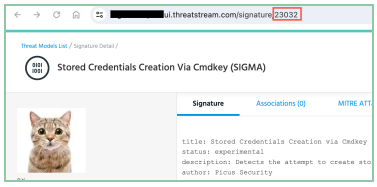
```
|erex ip in='82.247.137.135,108.32.74.68,92.239.20.53' not='1.1'
output
```

evalrule

Use the `evalrule` operator to evaluate sigma rule from ThreatStream on demand.

Syntax

```
|evalrule rule_id=<id>
```

`rule_id=<id>`	Specify the ThreatStream rule ID. **To find the rule ID in ThreatStream:** 1. Go to **Analyze** > **Threat Model**. 2. Select **Signatures** > **Sigma**. 3. Click the rule you want to go to the details page. 4. Note the rule ID as the last part of the URL.  **Note**: If you are a ThreatStream and Search customer, you can also initiate rule evaluation from ThreatStream. From the sigma rule details page, click **Actions** and select **Evaluate Rules**. For details, refer to ThreatStream online help.

Usage

The `evalrule` operator can be used with table visualizations to show matching results.

The `evalrule` operator enriches the results with the following fields:

- `rule_id_`
- `rule_name`
- `rule_level`
- `rule_status`
- `rule_tags`

The following Sigma rule fields correspond with Security Analytics event search fields.

Sigma Rule Fields	Event Search Fields
c-uri-query	url
c-useragent	user_agent
CommandLine	command_line
DestinationHostname	dest
DestinationIp	dest_ip
DestinationPort	dest_port
Details	reg_value_data
dst_ip	dest_ip
dst_port	dest_port
EventID	event_id
EventName	event_name
FileName	file_name
FilePath	file_path
Image	image
NewProcessName	image
OriginalFileName	original_file_name
ParentCommandLine	parent_command_line
ParentImage	parent_image
ParentProcessID	parent_process_id
Platform	platform
ProcessCommandLine	command_line
ProcessID	process_id

Sigma Rule Fields	Event Search Fields
SourceImage	parent_image
SourcePort	src_port
TargetFilename	file_name
TargetObject	reg_key
UserAgent	user_agent

For example, if the Sigma rule is written to evaluate `TargetFilename` or `FileName`, Match evaluates the event log `file_name` field. Contact Anomali to request support for additional fields.

Examples

Example 1: Evaluate and apply a detection rule.

```
|evalrule rule_id=35093
```

Example 2: Apply rules to filter matching events.

```
|where sourcetype = 'endpoint' | evalrule rule_id=35094
```

Example 3: Evaluates OCSF events against detection rule 4038 and appends the matched results to each event.

```
ocsf | evalrule rule_id=4038
```

eventstats

Use the `eventstats` operator to perform an aggregation function on a specified event log field and add a new column to the results table with the aggregation function results.

> **Caution:** Anomali recommends using the `eventstats` command on shorter time periods. Searches running over very large time ranges using `eventstats` can result in searches timing out.

Syntax

```
eventstats <aggr expression>
```

<aggr expression>	An aggregation function that takes an event log field as an argument. See "aggr" on page 436 for details about aggregation functions.

Usage

Use multiple pipes to perform multiple `eventstats` operations in a single query. The results include as many new columns as generated by the operations. Refer to the examples.

This operator works with both `eventlog` and `ocsf` schemas. For OCSF fields, use dotted notation to reference nested fields. Examples include `metadata.product.name` or `actor.user.name`.

> **Note:** If you do not use `as <newfield>` to specify an aggregation field name, the search results table generates columns names similiar to `aggr_col1`, `aggr_col2`, and so on.

Examples

Example 1: Eventlog with multiple aggregations

The following example creates two new columns that represent the total count of a field and another aggregate that calculates the max of another field:

```
|eventstats count(sourcetype) as total |eventstats max(bytes_in)
as maximum
```

Example 2: OCSF with statistical outlier detection

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Calculate mean and standard deviation for event counts, then identify outliers:

```
ocsf | where metadata.product.name != '' | bin event_time span=1h
| aggr count as cnt by event_time, src_endpoint.ip | eventstats
avg(cnt) as mean_event_count, stdev(cnt) as stdev_event_count |
calc upper_threshold = mean_event_count + 2 * stdev_event_count |
calc lower_threshold = mean_event_count - 2 * stdev_event_count |
calc is_outlier = if(cnt > upper_threshold OR cnt < lower_
threshold, "yes", "no")
```

extract

Use the `extract` operator to extract fields (key-value pairs) from the `_raw` field by specifying delimiters.

Syntax

```
|extract [<fields>] kvdelim=<string>
pairdelim=<string> [maxchars=<integer>]
[multivalues={true, false}]
```

`[<fields>]`	Specify a comma-separated list of field names to extract from the `_raw` field.
`kvdelim=<string>`	Specify one or more characters used to separate the key from the value. If any of the delimiters appears in the value, the field will not be extracted.
`pairdelim=<string>`	Specify one or more characters used to separate the fields (key-value pairs).
`[maxchars=<integer>]`	Optional. Specify a maximum number of characters to consider from the `_raw` field. If this option is not specified, the implicit default is 10,000 characters.
`[multivalues={true, false}]`	Optional. Specifies whether multivalue fields should be created. If this option is not specified, the implicit default is `false`.

Usage

You can use the `extract` operator when the `_raw` field contains different fields in a specific format and want to analyze them.

In event logs, the `_raw` field may contain numerous key-value pairs in a specific format—for example, comma separated, semicolon separated, and so on.

Examples

Example 1: Extract key-value pairs using custom delimiters.

```
|extract fields='field1,field2' kvdelim='=' pairdelim=','
```

Example 2: Extract multiple fields with colon and pipe delimiters.

```
|extract fields='monitor,date,source' kvdelim=':=' pairdelim='|,'
```

Example 3: Extract and sort multiple fields.

```
|extract fields='timelimit,messages' kvdelim='=' pairdelim=',' |
sort by timelimit,messa
```

fields

Use the `fields` operator to either specify the fields to be returned or rename the fields. You can optionally add a plus (+) or minus (-) sign to respectively include or exclude a comma-separated list of additional fields in the result table.

Syntax

```
|fields <field> [as new_field] [, <field> [as new_
field], ...]
```

```
|fields + <field> [as new_field] [, <field> [as new_
field], ...]
```

```
|fields - <field> [as new_field] [, <field> [as new_
field], ...]
|select <field> [as new_field] [, <field> [as new_
field], ...]
```

Usage

Use the `fields` operator to filter the results set. If you plan to create visualizations, include `fields` necessary to show what you want to visualize.

> **Note:** The `select` operator is an alias of `fields`. In other words, the usage for `select` is exactly the same as the usage for `fields`.

Optionally use the plus (+) or minus (-) sign to include additional fields in your results set. If these signs are not used, only the fields mentioned after the `fields` operator are displayed as the results set.

> **Note:** If you use the plus (+) or minus (-) sign after the `fields` operator, you must include at least one field. Not doing so will throw an error.

Examples

Example 1: Select specific fields from results.

```
|fields event_time, src, image, command_line
```

Example 2: Include multiple fields in output.

```
|where dcid = "10" | fields dcid as UlinkID, sourcetype, bytes_
in, bytes_out
```

Example 3: Exclude unwanted fields from output.

```
|where dcid = "10" | select dcid as UlinkID, sourcetype, bytes_
in, bytes_out
```

Example 4: Reorder and select event columns.

```
|bin event_time span = 1h | where sourcetype contains_ci * and
isnotnull(bytes_in) | fields + bytes_in as BytesIn, event_size as
Size, category as Catg
```

Example 5: Select and display renamed fields.

```
|where sourcetype contains_ci * | fields + bytes_in, event_size,
category
```

Example 6: Filter to selected fields only.

```
|where sourcetype != 'null' | fields + bytes_in, event_size,
category | fields - action, dcid, dest
```

Example 7: Filters out events with no product name, then returns only the timestamp, activity type, and source hostname fields.

```
ocsf | where metadata.product.name!='' | fields event_time,
activity_name, src_endpoint.hostname
```

fieldsummary

Use the `fieldsummary` operator to generate a summary of the output fields from the AQL.

Syntax

```
|fieldsummary [maxvals=<integer>]
[samplesize=<integer>] <fields>
```

`maxvals=<integer>`	Specify the maximum number of values to be included in the summary for each field. It must be a value between `2` and `100`. Default: `25`
`samplesize=<integer>`	Specify the maximum number of rows to include in the summary. It must be a value between `1` and `100000000` (100 million rows). The larger the sample size, the longer it will take. Default: `1000000` (1 million rows).
`<fields>`	Specify the list of fields to be included in the summary. If it is not specified, the operator will generate the summary for all fields. In the case of the `eventlog` table, by default, the operator generates the summary for the following fields: `sourcetype`, `dcid`, `src`, `action`, `dest`, `url`, and `user`. The following fields are excluded for the `eventlog` table: • `event_time` • `timestamp` • `message` • `original_event_time` • `volume` • `event_meta_id` • `_raw_id` • `day` • `hour` • `minute`

Usage

You can use the `fieldsummary` operator to generate statistics of the resulting fields.

This operator works with both `eventlog` and `ocsf` schemas. For OCSF fields, use dotted notation to reference nested fields. For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Examples include `metadata.product.name` and `src_endpoint.ip`.

> **Note:** When querying OCSF data, `fieldsummary` generates results for each OCSF class separately. This helps differentiate statistics across different event types within your OCSF data.

The summary includes the following columns for each field:

- `field_name`: Name of the field.
- `count`: Number of values in the field.
- `distinct_count`: Number of distinct values in the field if it is categorical (string or boolean) or it is `src_port` or `dest_port`.
- `max`: Maximum value in the field if it is numeric.
- `min`: Minimum value in the field if it is numeric.
- `mean`: Mean value for the field if it is numeric.
- `values`: List of the different values with their respective counts. It is an approximation of the most frequent values. This applies only for categorical fields (string or boolean) as well as `src_port` or `dest_port`.

> **Note:** The AQL query must specify a lookup table before this piped operator. It can be any lookup table.

Examples

Example 1: Eventlog field summary

```
example_lookup_table | where src_ip != '' and src_ip != 'null' |
fields src_ip , event_size , action | fieldsummary src_ip, event_
size, action
```

Example 2: OCSF field summary

```
ocsf | fieldsummary maxvals=5 metadata.product.name, activity_
name, src_endpoint.ip
```

filldown

Use the `filldown` operator to fill empty cells in the table with the value in the row above it from the same column.

Syntax

`|filldown`

Usage

You can use the `filldown` operator when you want to fill empty cells in the table with the value in the row above it from the same column. If there is no non-null value in the rows above the cell in the same column, the cell remains null.

You can use the `filldown` operator for data preprocessing to address gaps in data continuity. The `filldown` operator is useful when you have time series data with intermittent missing values or null fields.

Examples

Example 1: Propagate previous values to fill empty fields.

```
|filldown
```

fillnull

Use the `fillnull` operator to specify your preferred string or number to appear in results when a field has a null value.

Syntax

`|fillnull [value=<string|number>] [<field-list>]`

Usage

The `fillnull` operator fills fields that have a null value with the specified new value.

If no value is specified, the defaults are used: `"NULL"` for string field and `0` for integer fields.

Example

Example 1: Replace null field values with defaults.

```
|fields action, src, src_port , dest, dest_port, bytes_in, bytes_
out

|fillnull src value="no source"

|fillnull bytes_in, bytes_out
```

Example 2: Filters OCSF events by severity, selects network flow fields, caps at 100 rows, replaces null interface names with descriptive defaults and null byte counts with zero, then sorts by most recent event first.

```
ocsf | where severity in ('alert', 'info') | fields severity,
src_endpoint.interface_name, src_endpoint.port, dst_
endpoint.interface_name, dst_endpoint.port, traffic.bytes_in,
traffic.bytes_out | limit 100 | fillnull src_endpoint.interface_
name value='no source' | fillnull traffic.bytes_in,
traffic.bytes_out | fillnull dst_endpoint.interface_name
value='empty_dest' | sort event_time desc
```

fit

Use the `fit` operator to train a machine learning model on AQL search results.

Components

Typically, a machine learning model consists of the following components:-

`algorithm`	Specifies the name of the machine learning algorithm. Supported algorithms include: • ARIMA • birch • DBSCAN • DecisionTreeRegressor • DensityFunction • IsolationForest • LinearRegression • LocalOutlierFactor • LogisticRegression • OrdinalEncoder • RandomForest • StandardScaler
`option_name`	Indicates the model parameters. The format is as follows: `option_name=value` Refer to the documentation and examples provided for the particular supported algorithm.
`field_to_predict`	Includes the target field name. Required for supervised models only. Not applicable to unsupervised algorithms.
`explanatory_fields`	Captures the input fields used to train the model. Required for supervised models. Not used for unsupervised models.

`kfold_cv`	Ensures k-fold cross validation (if the algorithm supports it). When specified, the validation result is returned. Substitute a number of folds for *n* (minimum value is 2). Note: This option cannot be used together with `into` option to save the model.
`into`	Use the `into` keyword to store the trained model as an artifact that can later be applied using the `apply` operator.
`'model_name'`	Denotes the name of the model artifact. The model name must be wrapped in single or double quotation marks.

Usage

The following AQL piped operators are used to manage machine learning models.

AQL Operator	Usage
fit	Train a machine learning model on AQL search results.
"apply" on page 451	Compute predictions for the current search results based on a model that was learned using the `fit` operator.
"listmodels" on page 598	Return a list of machine learning models that were learned using the `fit` operator.
"deletemodel" on page 524	Delete a model that was generated by the `fit` operator.
"ml_model_summary" on page 614	Retrieve metadata about a model that was generated by the `fit` operator.

Examples

Example 1: Fit a linear regression model to data.

```
|fit LinearRegression species from petal_length petal_width
sepal_length sepal_width into 'iris_linear_regression_model'
```

Example 2: Build a classification model.

```
|fit LinearRegression species from petal_length petal_width
sepal_length sepal_width kfold_cv=5
```

Example 3: Apply trained model to new data.

```
|fit LogisticRegression species from petal_length petal_width
sepal_length sepal_width into 'iris_logistic_regression_model'
```

Example 4: Generate predictions from model.

```
|fit LogisticRegression species from petal_length petal_width
sepal_length sepal_width kfold_cv=5
```

fit ARIMA

Use the `ARIMA` algorithm when you want to fit a model using the statsmodels ARIMA (Auto-Regressive Integrated Moving Average) method for forecasting time series data.

This operator also supports seasonal ARIMA to provide reliable forecasting patterns for data with seasonal intervals such as daily, weekly, or monthly intervals through the `seasonal_order` parameter explained in the "Usage" on page 544 section. The model, therefore, captures recurring patterns such as:

- Daily cycles (for example, login peaks during business hours)
- Weekly cycles (for example, weekday versus weekend network traffic)
- Other periodic behaviors (for example, batch jobs or month-end activity)

When you apply the model, all original observations are retained and the results include actual values and predictions in separate

columns, helping you more readily identify where the model forecast deviates from reality.

See the official statsmodels ARIMA documentation for more information.

Syntax

```
fit ARIMA [arima_order='<p>-<d>-<q>'] [seasonal_
order='<P>-<D>-<Q>-<s>'] [forecast_k=<forecast_
steps>] [conf_interval=<confidence>]
[holdback=<exclude_from_training>] <value_field>
from <time_field>
```

`<p>-<d>-<q>`	(Optional) Non-seasonal ARIMA parameters that determines the complexity of the model, where: • p – Autoregressive (AR) order • d – Differencing order • q – Moving average (MA) order Default=`'1-1-1'`
`<P>-<D>-<Q>-<s>`	(Optional) Seasonal ARIMA parameters P-D-Q-s, where: • P=seasonal AR order, • D=seasonal differencing, • Q=seasonal MA order, • s=seasonal period for example, 24 for hourly data with daily seasonality, 7 for daily data with weekly seasonality) Example: `seasonal_order='1-1-1-24'` denotes hourly data with daily seasonality. `seasonal_order='0-1-1-168'` denotes weekly seasonality on hourly data (24 × 7).

`forecast_steps`	(Optional) Number of future time steps to forecast, that is, how far into the future the model will predict. For example, `forecast_k=24` on hourly data forecasts the next 24 hours. Default: `0`
`confidence`	(Optional) Confidence interval width (in percent) around the forecast values. For example, `conf_interval=95` creates 95% upper and lower confidence bounds, signifying that there is a 95% probability that the actual future values will be within the calculated confidence interval around the forecasted values. Default: `95`
`exclude_from_training`	(Optional) Number of most recent observations to exclude from model training and keep for backtesting. For example, if you have 100 data points and set holdback to 10, the last 10 data points will not be used in training the model. Default: `0`
`value_field`	Required. The numeric field containing the time-series values to model and forecast, for example, event counts, login counts, or bytes transferred.
`time_field`	Required. The time field that defines the ordering of the series.

Usage

You can use seasonal ARIMA with `seasonal_order` to:

- Forecast time-series metrics that exhibit recurring patterns (daily, weekly, or other cycles).
- Reduce false positives by distinguishing normal seasonal spikes from genuine anomalies.

Common real-world use cases include:

- Authentication logs: Business-hours peaks and overnight troughs.
- Network traffic: Different traffic levels during weekdays versus weekends, or day versus night.
- Batch jobs and reports: Periodic spikes from scheduled tasks.

Examples

Example 1: Learns how `dest_port` has changed over time, and predicts the next 100 time points of `dest_port`, with a 90% confidence band for each prediction.

```
| fit ARIMA forecast_k=100 conf_interval=90 dest_port from event_
time
```

Example 2: Models hourly Azure AD login counts over the past seven days, capturing weekly seasonality with a seven-hour cycle on hourly data.

```
| where sourcetype ='azure_ad'

| where event_time earliest=-7d, latest=-1h timetrend
columnname=timestamp count as log_count span=1h

| fit ARIMA forecast_k=24

          arima_order='2-1-2'

          seasonal_order='1-1-1-7'

          conf_interval=95

          holdback=24

          log_count from timestamp
```

Example 3: Forecasts the next three hourly event count values using SARIMAX, trained on the hourly event trend derived from OCSF data.

```
ocsf | timetrend count as test span=1h | predict test SARIMAX
forecast_k=3
```

fit birch

Use the `birch` algorithm when you want to divide data points into separate clusters using the scikit-learn BIRCH (balanced iterative reducing and clustering using hierarchies) algorithm.

Syntax

```
fit birch [option_name]=[value] [into '<model_
name>']
```

`[option_name]=[value]`	Specify model parameters: • `k=<int>`: Specify the total number of clusters into which the data is segmented after the concluding clustering process, during which the subclusters originating from the CF tree's leaves are treated as fresh samples. • `partial_fit={true|false}`: Specify whether an existing model should be updated incrementally or not. It enables the modification of an existing model with new data alone, eliminating the need to retrain it on the entire training data set.
`into`	Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

Usage

You can use the `birch` algorithm to divide data points into separate clusters. The algorithm assigns each event to a cluster and records this assignment in a newly created field called "cluster". This algorithm supports incremental fitting.

For information about the BIRCH algorithm, refer to scikit-learn documentation:

https://scikit-learn.org/stable/modules/clustering.html#birch

Examples

Example 1: Cluster data using BIRCH algorithm.

```
|fit Birch k=3 from petal_length petal_width sepal_length sepal_
widths
```

Example 2: Apply hierarchical clustering with parameters.

```
|fit Birch k=3 from petal_length petal_width sepal_length sepal_
width into "birch_model"
```

fit DBSCAN

Use the `DBSCAN` algorithm when you want to fit a model to cluster the data using the scikit-learn DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm.

Syntax

`fit DBSCAN [option_name]=[value] <fields>`

`[option_name]=[value]`	Specify model parameters. The following model parameters are supported: • `eps={float}, default=0.5` • `min_samples={integer}, default=5` • `metric={string}, default=euclidean` • `metric_params={dict}, default=None` • `algorithm={'auto', 'ball_tree', 'kd_tree', 'brute'}, default='auto'` • `leaf_size={integer}, default=30` • `p={float}, default=None` For information on the model parameters, refer to scikit learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
`<fields>`	Input fields for clustering.

Usage

You can use the `DBSCAN` algorithm to fit a model that clusters the value of numeric fields using the scikit-learn DBSCAN algorithm. The input training data has to be preprocessed, encoded, and normalized if required before passing to this algorithm.

Examples

Example 1: Apply DBSCAN clustering algorithm.

```
|fit dbscan sepal_length sepal_width petal_length petal_width
species
```

Example 2: Cluster data with custom parameters.

```
|fit dbscan eps=0.6 algorithm='ball_tree' sepal_length sepal_
width petal
```

fit DecisionTreeRegressor

Use the `DecisionTreeRegressor` algorithm when you want to fit a model that predicts a continuous variable using the scikit-learn decision tree regression estimator. When applying the model, predictions are appended to the full dataset, allowing you to directly compare actual and predicted values.

Syntax

```
fit DecisionTreeRegressor [option_name]=[value]...
<field_to_predict> from <explanatory_fields> [kfold_
cv=<enter_max_cross-val_fields>] [into '<model_
name>']
```

`[option_name]=[value]`	Specify model parameters: • `criterion={'squared_error'| 'friedman_mse'|'absolute_error'| 'poisson'}, default='squared_error'` • `splitter={'best'|'random'}, default='best'` • `max_depth={int}, default=10` • `min_samples_split={int|float}, default=2` • `min_samples_leaf={int|float}, default=1` • `min_weight_fraction_leaf={float}, default=0.0` • `max_features={int|float|'sqrt'| 'log2'}, default=None` • `random_state={int}, default=None` • `max_leaf_nodes={int}, default=None` • `min_impurity_decrease={float}, default=0.0` • `multi_class={'auto'|'ovr'|'multinomial'}` • `ccp_alpha={non-negative float}, default=0.0` For information on the model parameters, refer to the official Scikit-learn documentation.
`<enter_max_tree_depth>`	Specify the maximum depth of the decision tree. Providing the depth preempts the model from growing too large and overfitting, ensuring reliable results when visualizing the tree. Default value: `10`. **Note:** `max_depth` must be the first parameter in the query.
`<field_to_predict>`	Specify the target field name.

`<explanatory_fields>`	Specify input fields used to train the model.
	Note: When applying the model, the name of the explanatory fields need to be exactly the same as the one used to create the models. Any deviation from the name may lead to errors.
	For example:
	`eventlog | where event_id='4624'`
	`| bin event_time span=1h`
	`| aggr count() as logon_count by event_time, user`
	`| fit DecisionTreeRegressor max_depth=5 logon_count from event_time into 'logon_baseline_model'`
	`| apply logon_baseline_model as predicted_logon_count event_time`

`<enter_max_cross-val_fields>`	Specify this option to support k-fold validation. When specified, the validation result will be returned. Substitute a number of folds for *n*. The number of folds must be at least two. **Note:** This option cannot be used together with the `into` option to save the model.
`into`	Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

Usage

Use `DecisionTreeRegressor` to predict a continuous numeric value from historical security data. For example, you can determine the expected logon counts per time window or model typical outbound bytes for a connection. The model learns decision rules from labeled training data and appends predictions alongside actual values when applied, making it easy to spot deviations.

This approach ensures DecisionTreeRegressor is:

- Effective for understanding normal behavior and flagging deviations in numeric security metrics.
- Interpretable, that is, the resulting tree structure can be visualized to understand which features drive predictions.

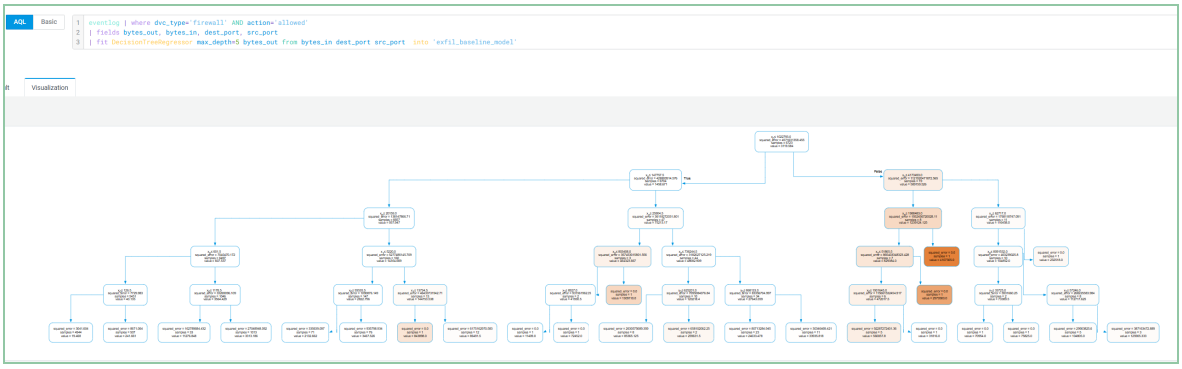
See "About the Decision Tree Visualization Type " on page 277 for more information on how to visualize decision trees.

Examples

Example 1: Trains a decision tree on allowed firewall traffic to expected outbound bytes from inbound volume and port pair, for detecting data exfiltration anomalies.

```
eventlog | where dvc_type='firewall' AND action='allowed'
| fields bytes_out, bytes_in, dest_port, src_port
| fit DecisionTreeRegressor max_depth=5 bytes_out
  from bytes_in dest_port src_port
  into 'exfil_baseline_model'
```

After the search is complete, use the Decision Tree visualization to visualize your results. The query above produces a decision tree visualization to identify ambiguous traffic patterns. To view the tree, switch to the **Visualization** tab on the Search page, as follows:



Example 2: Buckets IAM logon events hourly, then trains a decision tree to baseline expected logon frequency per time window for insider threat detection.

```
ocsf
| where category_name='Identity & Access Management'
  AND activity_name='Logon'
| bin event_time span=1h
| aggr count() as logon_count by event_time, actor.user.name
| fit DecisionTreeRegressor max_depth=5 logon_count
  from event_time into 'logon_baseline_model'
```

fit DensityFunction

Use the DensityFunction algorithm to model normal historical behavior using Kernel Density Estimation (KDE) and generate anomaly scores for new data.

Syntax

Model fitting phase

```
| fit densityfunction [bandwidth] [threshold]
[lower_threshold] [upper_threshold] <numeric_field>
[by <group_field> [, <group_field>...]] [into
'<model_name>']
```

`bandwidth`	The smoothing factor for the Kernel Density Estimation, controlling the smoothness or extent of details of the density curve. Lower bandwidth: sharper, more sensitive density curve (detects small variations). Higher bandwidth: smoother, more generalized curve (ignores small fluctuations).

threshold	A general anomaly detection cutoff. Any density score below 0.01 is usually considered anomalous.
lower_threshold	A lower bound for extremely rare events. This helps separate critical anomalies from normal low-density variations.
upper_threshold	A high-density threshold. Useful for validation, debugging, or categorizing events as strongly normal.
numeric_field	Field whose distribution you want to model.
by <group_field>	Build separate density models per group. For example, one model per sourcetype.
into '<model_name>'	Saves the trained model under this name so apply can use it later. Must be a string without spaces, wrapped with single or double quotes.

Model apply phase

```
| apply <model_name> as <output_field> <numeric_field> <group_field>[, <group_field>...]
```

<model_name>	Name of a previously saved DensityFunction model.
as <output_field>	Name of the field that will hold the score.
<numeric_field>	The same field that was used during the fit phase.
<group_field>	The same grouping fields used during the fit phase.

How DensityFunction Works

Training Phase (Model Fitting)

During the fitting phase, DensityFunction analyzes historical data and learns a probability density function that represents “normal” behavior. The function:

- Accepts one or more numeric fields as training input.
- (Optional) accepts grouping fields to build separate baseline models for each user, host, device, or other entity.
- Uses KDE with a Gaussian kernel to generate a smooth density curve from raw observations.
- Allows configuration of key parameters such as:
  - Bandwidth: controls how smooth or detailed the density curve is.
  - Kernel type: by default, Gaussian is the kernel type.
  - (Optional) Sampling options: synthetic sampling from the density curve.
  - Random seed: ensures reproducibility.

Once training is complete, you can optionally save the model and reused it to score new data.

Apply Phase (Scoring)

In the apply phase, the trained DensityFunction model:

- Evaluates each new incoming event.
- Determines whether the event fits within the learned normal distribution.
- Generates a density score indicating how typical or unusual the new event is:
  - Higher density indicates a highly expected or normal outcome.
  - Lower density indicates a rare or anomalous outcome.
- This score can be used to:
  - Trigger alerts
  - Feed dashboards
  - Power downstream detection logic
  - Enhance contextual analysis

When grouping fields is used during training, new data is automatically routed to the appropriate group-specific model.

Usage

DensityFunction is designed for scenarios where you need to:

- Understand normal behavior in your environment over time.
- Detect anomalies such as spikes, dips, or unusual distributions.
- Capture cyclical patterns, including:
  - Hourly authentication activity
  - Daily or weekly access patterns
  - Seasonally recurring event frequencies
- Identify outliers that may indicate security issues, misconfigurations, or system failures.

DensityFunction is particularly effective when analyzing cyclical or periodic security activity, such as login behavior by hour of day, network access patterns by weekday, or repeated operational workflows that follow predictable rhythms.

Usage Recommendation

- Ensure consistent lengths of input-output fields, valid JSON, and avoid empty datasets.
- KDE requires enough data per group. Very small groups lead to poor density estimation.
- Avoid grouping fields with extremely high cardinality unless necessary.
- Ensure numeric training fields are properly normalized or aggregated.

Performance considerations

- KDE for 1 model is fast, but KDE for thousands of groups becomes heavy.
- 100k groups is feasible possible but performance might degrade.
- Performance might degrade exponentially with:
  - More grouping fields
  - High cardinality values (for example, user IDs)
  - Large datasets

Example

Example 1: In the following example, the model learns normal event-count patterns per sourcetype over 10-minute intervals and then scores new intervals to identify whether each count is typical or anomalous.

Model fitting phase

```
| bin event_time span=10m | aggr count as cnt by event_time,
sourcetype

| fit densityfunction cnt by sourcetype into 'den_func_model'
```

Model apply phase

```
| bin event_time span=10m | aggr count as cnt by event_time,
sourcetype

| apply den_func_model as final_densi
```

fit IsolationForest

Use the `IsolationForest` operator to catch anomalous behavior, based on an unsupervised anomaly-detection algorithm optimized for high-volume event data. Instead of profiling normal behavior, it focuses on isolating anomalies. Because outliers are rare and different, they require fewer random splits in a tree structure to isolate, leading to a shorter average path length.

Syntax

```
fit IsolationForest [option_name=value] <fields_to_
use> [into '<model_name>']
```

`[option_name]=[value]`	Specify model configuration parameters. All scikit-learn parameters are supported (e.g., n_estimators, contamination, bootstrap, random_state). • `n_estimators` • `contamination` • `bootstrap` • `random_state` For information on the model parameters, refer to scikit learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html
`<fields_to_use>`	One or more numeric fields used as model input.
`into`	Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

How IsolationForest Works

The IsolationForest model builds a collection of random isolation trees:

1. A random feature is selected.
2. A random split point for that feature is chosen.
3. The dataset is recursively partitioned until each observation is isolated.

Anomalous points, being easier to isolate, will have shorter average path lengths across the forest.

The model outputs two key values:

- Anomaly Score: A continuous severity score, wherein a higher score indicates a more anomalous record.
- Prediction: A binary label indicating if the record is anomalous or not.

These values can then be used for alerting, dashboarding, and downstream investigations.

Usage

> **Note:** The model will be deleted if it is not used for 90 days.

You can use the `IsolationForest` algorithm to isolate anomalies. Because outliers are rare and different, they require fewer random splits in a tree structure to isolate, supporting you in anomaly detection within AQL workflows.

This approach makes IsolationForest:

- Effective for detecting unusual spikes or dips in event counts
- Scalable to large security datasets
- Useful for catching rare but meaningful deviations in behavior

Usage Recommendation

- IsolationForest does not understand random strings, JSON, or high-cardinality categorical fields unless appropriately transformed.
- Models trained on fewer than a few hundred rows may produce unstable results.
- IsolationForest detects statistical anomalies, not malicious intent. Analysts should, therefore, confirm anomalies before escalating.
- High anomalies may still be benign (expected spikes).
- Use lower contamination values to avoid noise. 0.01–0.05 is typical for security datasets.

Examples

The follwing example trains an IsolationForest model on user activity and MFA-failure patterns to identify statistically unusual authentication behavior.

Step 1: Train a model on aggregated event counts

```
| aggr count as cnt by sourcetype

| fit IsolationForest n_estimators=50 contamination=0.2 random_
state=42 bootstrap=true cnt
```

Step 2: Train and save the model

```
| aggr count as cnt by sourcetype

| fit IsolationForest n_estimators=50 contamination=0.2 random_
state=42 bootstrap=true cnt into "iso_model"
```

Step 3: Apply the saved model

```
| aggr count as cnt by sourcetype

| apply iso_model cnt
```

fit LinearRegression

Use the `LinearRegression` algorithm when you want to explore how one or more independent variables relate to a dependent variable, by fitting a model that predicts the value of numeric fields and evaluating the fit of the regression model.

Furthermore, you can also use the Linear Regression Visualization Type, on the AQL Search page, to visualize the results of your regression model.

Syntax

```
fit LinearRegression [option_name]=[option_value]...
<field_to_predict> from <explanatory_fields> [kfold_
cv=n] [into '<model_name>']
```

`[option_name]=[value]`	(Optional) Specify model parameters: • `fit_intercept={true|false}` • `Positive={true|false}` For information on the model parameters, refer to scikit learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
`<field_to_predict>`	Specify the target field name.
`<explanatory_fields>`	Specify input fields used to train the model.
`[kfold_cv=n]`	(Optional) Specify this option to support kfold validation. When specified, the validation result will be returned. Substitute a number of folds for *n*. The number of folds must be at least 2. Note: This option cannot be used together with `into` option to save the model.
`into`	(Optional) Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

Usage

You can use the `LinearRegression` algorithm to fit a model that predicts the value of numeric fields. The input training data has to be preprocessed, encoded, and normalized if required before passing to the fit operator. This algorithm supports the `kfold` cross-validation option.

Examples

Example 1: Visualize single-variable linear regression

```
| fit OrdinalEncoder action src | fit LinearRegression encoded_
action from encoded_src into 'linear_regression_model'
```

Example 2: Visualize single-variable linear regression with residuals

```
| fit OrdinalEncoder action src | fit LinearRegression encoded_
action from encoded_src into 'linear_regression_model2' with_
residuals=true
```

Example 3: Visualize multiple linear regression

```
| fit OrdinalEncoder action src src_port | fit LinearRegression
encoded_action from encoded_src encoded_src_port into 'linear_
regression_model3'
```

fit LocalOutlierFactor

Use the `LocalOutlierFactor` algorithm when you want to identify anomalous data points using the scikit-learn Local Outlier Factor (LOF).

Syntax

```
fit LocalOutlierFactor [option_name]=[value]
<fields> [into <model name>]
```

`[option_name]=[value]`	Specify model parameters. The following model parameters are supported: • `n_neighbors=<integer>` • `algorithm={'auto', 'ball_tree', 'kd_tree', 'brute'}` • `leaf_size=<integer>` • `metric=<string>` • `pfloat=<float>` • `contamination={auto|<float>}` • `novelty={true|false}` For information on the model parameters, refer to scikit learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html
`<fields>`	Specify input fields used to train the model.
`into`	Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

Usage

You can use the `LocalOutlierFactor` algorithm to identify anomalous data points using the scikit-learn Local Outlier Factor (LOF). This algorithm measures the local density deviation of a data point with respect to its neighbors. In other words, the algorithm assesses how isolated or different a data point is compared to its local neighborhood. The key idea is that outliers often have a significantly lower density of neighboring points than the majority of the data. `LocalOutlierFactor` performs one-shot learning and is limited to fitting on training data and returning outliers.

Examples

Example 1: Detect anomalies using LocalOutlierFactor.

```
|fields petal_length, petal_width, sepal_length,

sepal_width, species <br />|fit LocalOutlierFactor petal_length

petal_width sepal_length sepal_width species
```

Example 2: Identify outliers in multivariate data.

```
|fields petal_length, petal_width, sepal_length,

sepal_width, species <br />|fit LocalOutlierFactor n_neighbors=10

with_std=true petal_length petal_width sepal_length sepal_width
species
```

Example 3: Score observations for anomaly detection.

```
|fields petal_length, petal_width, sepal_length,

sepal_width, species <br />|fit LocalOutlierFactor novelty=true
wit
```

fit LogisticRegression

Use the `LogisticRegression` algorithm when you want to fit a model to predict the value of categorical fields using the scikit-learn logistic regression estimator.

Syntax

```
fit LogisticRegression [option_name]=[value]...
<field_to_predict> from <explanatory_fields> [kfold_
cv=n] [into '<model_name>']
```

`[option_name]=[value]`	Specify model parameters: • `penalty={'l1'|'l2'|'elasticnet'|none}` • `dual={true|false}` • `tol=<float>` • `C=<float>` • `fit_intercept={true|false}` • `intercept_scaling=<float>` • `class_weight={'balanced'|None}` • `random_state=<integer>` • `solver={'lbfgs'|'liblinear'|'newton-cg'|'newton-cholesky'|'sag'|'saga'}` • `max_iter=<integer>` • `multi_class={'auto'|'ovr'|'multinomial'}` • `warm_start={true|false}` • `l1_ratio=<integer>` For information on the model parameters, refer to scikit learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
`<field_to_predict>`	Specify the target field name.
`<explanatory_fields>`	Specify input fields used to train the model.

`[kfold_cv=n]`	Specify this option to support kfold validation. When specified, the validation result will be returned. Substitute a number of folds for *n*. The number of folds must be at least 2. Note: This option cannot be used together with `into` option to save the model.
`into`	Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

Usage

You can use the `LogisticRegression` algorithm to fit a model to predict the value of categorical fields using the scikit-learn logistic regression estimator. The input training data has to be preprocessed, encoded, and normalized if required before passing to the fit operator. This algorithm supports the `kfold` cross-validation option.

Examples

Example 1: Train a logistic regression classifier.

```
|fields petal_length, petal_width, sepal_length, sepal_width,
species <br />|fit LogisticRegression species from petal_length
petal_width sepal_length sepal_width into 'iris_logistic_
regression_model'
```

Example 2: Build classification model.

```
|fields petal_length, petal_width, sepal_length, sepal_width,
species <br />|fit LogisticRegression species from petal_length
petal_width sepal_length sepal_width kfold_cv=5
```

Example 3: Apply logistic regression for predictions.

```
|fields petal_length, petal_width, sepal_length, sepal_w
```

fit OrdinalEncoder

Use `OrdinalEncoder` to encode categorical features into integer-encoded vector representations.

Each unique category is mapped to a distinct integer, producing a compact numeric form that can be consumed by downstream analytics, detection logic, or machine learning models.

Syntax

```
| fit OrdinalEncoder [handle_unknown={'error'|'use_
encoded_value'}] [unknown_value=<int>] [encoded_
missing_value=<int>] <categorical_fields> [into
'<model_name>']
```

`handle_unknown`	(Optional) Controls how the encoder behaves when it encounters a category during transformation that was not observed during fitting. `use_encoded_value`: Default value. Unknown categories are mapped to a predefined integer. `error`: Transformation fails if an unknown category is encountered.
`unknown_value`	(Optional) Specifies the integer value used to encode previously unseen categories. Default value is -1.
`encoded_missing_value`	(Optional) Specifies the integer value used to encode missing or null categories. Default value is -2.
`categorical_fields`	Each field is encoded independently, producing one integer column per input field.
`into '<model_name>'`	Persists the learned category-to-integer mapping into a named model object.

Usage

You can use `OrdinalEncoder` to encode categorical features as an vector of integers. It is particularly useful when you need a stable, low-dimensional

representation of categorical data.

As an example, in cybersecurity workflows, ordinal encoding can be applied to fields such as:

- `action`: ALLOW, DENY, BLOCK
- `priority`: low, medium, high
- `event_type`: login, file access, process start
- `authentication_method`: password, token, certificate
- `device_type`

In terms of real-world applications, it is especially effective for:

- UEBA feature pipelines
- Event classification
- Alert enrichment
- Log normalization prior to ML scoring

How OrdinalEncoder works

Ordinal encoding assigns arbitrary integer IDs to categories based on the order in which categories are encountered during fitting.

- It creates a consistent numeric mapping for categorical values.
- It preserves category identity, not semantic distance.
- It does not imply ranking, magnitude, or priority between categories, unless the original featurse are inherently ordinal.
- The transformation does not enfore semantic ordering unless the original features are truly ordinal, that is, they have inherent ordering.

For example, the following encoding does not mean BLOCK > DENY > ALLOW in severity or importance:

```
ALLOW = 0
```

```
DENY = 1

BLOCK = 2
```

However, an inherent order *does exist* in the following encoding for high priority compared to low priority since priority has inherent ordering:

```
low = 0

medium = 1

high = 2
```

Such encoding helps tree-based models such as IsolationForest to make meaningful decision splits like `priority >= medium`, but should not be used for distance-based models to assume numeric spacing for arithmetic expressions such as `high - medium == medium - low`.

Examples

```
| fit ordinalencoder handle_unknown='use_encoded_value' unknown_
value=-2 encoded_missing_value=-1 action
```

```
| fit ordinalencoder action into 'model_name'
```

fit RandomForest

Use the RandomForest algorithm to classify security events by combining multiple decision trees into a single ensemble model using the Scikit-learn RandomForest method.

By aggregating predictions across many trees, Random Forest reduces overfitting and handles noisy, high-dimensional security data more robustly than a single decision tree. These capabilities make it suitable for applications like threat type prediction or malware detection.

Syntax

```
fit RandomForest [option_name]=[value]... <field_to_
predict> from <explanatory_fields> [kfold_cv=<enter_
max_cross-val_fields>] [into '<model_name>']
```

`[option_name] = [value]`	Specify model parameters: • `n_estimators=<integer>` • `criterion={'gini'|'entropy'}` • `max_depth=<integer>` • `min_samples_split=<integer|float>` • `min_samples_leaf=<integer|float>` • `min_weight_fraction_leaf=<float>` • `max_features={'sqrt', 'log2', None}` • `max_leaf_nodes=<integer>` • `min_impurity_decrease=<float>` • `bootstrap=={true|false}` • `oob_score=={true|false}` • `random_state={integer|<random state instance>|None}` • `verbose=<integer>` • `warm_start=={true|false}` • `class_weight={'balanced'|'balanced_subsample'|<dict or list of dicts>|None}` • `ccp_alpha=<non-negative float>` • `max_samples=<integer|float>` • `monotonic_cst={1|0|-1}` For information on the model parameters, refer to the official Scikit-learn documentation.

`<enter_max_tree_depth>`	Specify the maximum depth of the decision tree. Providing the depth preempts the model from growing too large and helps avoid overfitting, ensuring reliable results when visualizing the tree. Default value: `10`. **Note:** `max_depth` must be the first parameter in the query.
`<field_to_predict>`	Specify the target field name.

`<explanatory_fields>`	Specify input fields used to train the model.
	Note: When applying the model, the name of the explanatory fields need to be exactly the same as the one used to create the models. Any deviation from the name may lead to errors.
	For example:
	`eventlog | where dvc_type='firewall'`
	`| fields action, bytes_in, bytes_out, src_port, dest_port`
	`| fit RandomForestClassifier max_depth=5 n_estimators=100 action from bytes_in bytes_out src_port dest_port into 'firewall_action_classifier'`
	`| apply firewall_action_classifier bytes_in bytes_out src_port dest_port`

`<enter_max_cross-val_fields>`	Specify this option to support k-fold validation. When specified, the validation result will be returned. Substitute a number of folds for *n*. The number of folds must be at least two. Note: This option cannot be used together with `into` option to save the model.
`into`	Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

Usage

Use `RandomForestClassifier` to classify security events by training an ensemble of decision trees on labeled data. By aggregating votes across many trees, it reduces overfitting and handles noisy, high-dimensional security telemetry more robustly than a single decision tree.

This approach makes RandomForestClassifier:

- Well suited for binary and multi-class security classification tasks such as allowed-blocked, malicious-benign, or threat type classification.
- More resilient to irrelevant or noisy features than a single decision tree.
- Effective on high-volume eventlog and OCSF data.

See "About the Decision Tree Visualization Type " on page 277 for more information on how to visualize decision trees.

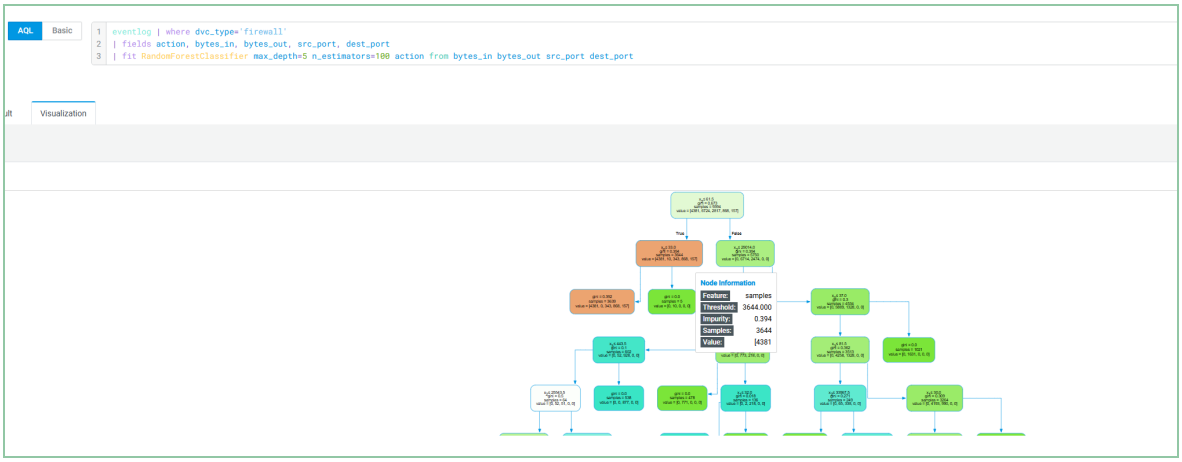
Examples

The following example trains a Random Forest classifier on firewall traffic to predict whether a connection is allowed or blocked, using byte volumes and port pairs as features.

```
eventlog | where dvc_type='firewall'
```

```
| fields action, bytes_in, bytes_out, src_port, dest_port

| fit RandomForestClassifier max_depth=5 n_estimators=100 action
from bytes_in bytes_out src_port dest_port
```

After the search is complete, use the Decision Tree visualization to visualize your results. The query above produces a decision tree visualization to identify ambiguous traffic patterns. To view the tree, switch to the **Visualization** tab on the Search page, as follows:



fit StandardScaler

Use the `StandardScaler` algorithm to standardize data fields to attain zero mean and unit standard deviation, using the scikit-learn standard scaler algorithm.

The StandardScaler operator retains all original columns, including those used for grouping and context. This enhancement ensures that contextual columns like `src_user` and `time_bucket_format` are included in the result set, allowing users to continue analysis seamlessly, with full access to both the scaled features and their associated context.

Syntax

```
fit standardscaler [option_name]=[value] <fields>
[into '<model_name>']
```

`[option_name]=[value]`	Specify model parameters: • `with_mean={true|false}` • `with_std={true|false}` For information on the model parameters, refer to scikit learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
`<fields>`	Input fields for scaling.
`into`	Specify the `into` keyword to store the learned model in an artifact that can later be applied to new search results with the `apply` operator.
`'<model_name>'`	Specify the name of the model to save as an artifact, so that it can be used with the `apply` operator. The model name must be wrapped with single or double straight quotation marks.

Usage

Use `StandardScaler` before applying machine learning algorithms that are sensitive to feature scales, especially those that use Euclidean distances. StandardScaler transforms features to have a mean of zero and standard deviation of one, ensuring all features contribute equally to the model's learning process.

Examples

Example 1:

```
|fit standardscaler petal_length petal_width sepal_length sepal_
width species
```

Example 2:

```
|fit standardscaler with_mean=true with_std=true petal_length
petal_width sepal_length sepal_width species
```

Example 3:

```
|fit standardscaler with_mean=true with_std=true petal_length
petal_width sepal_length sepal_width species into 'iris_std_
scaler'
```

Example 4:

```
| bin span=20m event_time as time_bucket

| calc time_bucket_format= strftime(time_bucket, "%Y-%m-%d
%H:%M")

| aggr count() as logon_count, count_distinct(src_ip) as dist_ip
by user, time_bucket_format

| fit standardscaler logon_count dist_ip
```

foreach

Use the `foreach` operator to iterate through a selection of fields and apply a subsearch to each field.

Syntax

`|foreach <field-list> [ <subsearch> ]`

`<field-list>`	A comma-separated list of fields to iterate through. The wildcard `*` specifies to iterate the subsearch over all fields.
`[ <subsearch> ]`	A `calc` command to be executed as the fields are iterated through. Subsearch can use a template value `<<FIELD>>` that is replaced by each field in the specified field list when the `calc` function is executed.

Usage

The `foreach` operator can be used to obtain results from across multiple fields in each row. This can be useful for calculating the sum or average of a selection of fields, for example.

Examples

Example 1: Iterate calculations across multiple tables.

```
|foreach test1, test2, test3 [ calc total = total + <<FIELD>> ]
```

Example 2: Apply operations to each table separately.

```
|foreach * [ calc <<FIELD>> = concat(<<FIELD>>, 'abc') ]
```

Example 3: Aggregates event counts by status, then counts those counts, then iterates over each of the `cnt` and `cnt2` fields adding the current value of the field to `cnt` in each iteration.

```
ocsf | aggr count as cnt by status |aggr count as cnt2 by cnt |
foreach cnt, cnt2 [ CALC cnt = cnt + <<FIELD>> ]
```

fromjson

Use the `fromjson` operator to extract the top level keys from the JSON object and create a table with keys as column names and values as field values.

Syntax

```
|fromjson <json_field> [prefix=<string>]
```

`<json_field>`	Specify a field that contains a JSON object.
`[prefix=<string>]`	Optional. Specify a prefix to append to extracted column names.

Usage

You can use the `fromjson` operator to extract fields from a JSON field. The top level keys are extracted.

Use the optional prefix argument when you want to append the custom string to the column names. In your AQL query syntax, you must specify the JSON field argument before the optional prefix argument.

Examples

Example 1: Parse JSON field and extract nested data.

```
|<a query that generates a json_field> |fromjson <json_field>
prefix="extr_"
```

Example 2: Convert JSON string to structured fields.

```
|test_table_fromjson| fromjson json_field1 prefix="foo_"
```

Example 3: Takes the last 10 OCSF events, packages `class_name` and `metadata.product.name` into a JSON object, then parses it back out into individual fields.

```
ocsf | tail 10 | fields class_name, metadata.product.name |
makejson class_name, metadata.product.name output=new_field |
fromjson new_field
```

gti

> **Note:** The gti operator uses the Google Threat Intelligence API key associated with the Google Threat Intelligence enrichment configured for your ThreatStream organization.

Use the `gti` operator to enrich observables with comprehensive threat intelligence using the ThreatStream Google Threat Intelligence enrichment, including detailed threat scores, verdict counts, community insights, and related indicators.

Syntax

```
| gti <observable>
```

`<observable>`	The field name containing the observable, enclosed in double quotes. Supported observable types include: • IP addresses • Domains • File hashes (MD5, SHA-1, SHA-256) • URLs

Usage

You can use the `gti` operator to enrich your observables with the latest Google Threat Intelligence verdicts and metadata, and then drive downstream detection,

investigation, and response actions. The operator is particularly useful when you need:

Consolidated GTI threat scoring

Use the `threat_score` and the per-engine verdict counts (`malicious`, `suspicious`, `harmless`, `undetected`) to quickly determine how strongly Google Threat Intelligence classifies an indicator as malicious. This is useful when you want a single numeric score along with its evidence to feed into correlation rules or risk models.

Community scoring and analyst insight

Use `community_malicious` and `community_harmless` to understand how the Google analysts and the broader security community view the IOC.

Using indicators for deeper investigation

Use `associated_ioc` and `other_hashes` to expand your investigation from a single IOC to related observables:

Recency checks

Use `last_analysis` to understand when Google Threat Intelligence last analyzed the IOC.

Examples

Example 1: Enrich a file hash (SHA-256) with Google Threat Intelligence data.

```
| gti
"f46890b4f94821b9c52f3c49e34591203f2ced37339713aaec5b9d65c2b8f3d"
```

Example 2: Enrich an IP address field from your data source with Google Threat Intelligence data.

```
| where scrip != null | fields + srcip | gti srcip
```

Example 3: Enrich a domain with Google Threat Intelligence data.

```
| gti "malicious-domain.com"
```

head

Use the `head` operator to show the first *n* results in the result set according to the sort order.

Syntax

```
|head [n]
```

Usage

Similar to the Unix `head` command.

For non-aggregation queries, if no sort order is specified, the `head` operator culls the first *N* results falling in the time range by `event_time`.

For aggregations, if no sort order is specified, the `head` operator draws the line at random.

Example

Example 1: Limit results to the first N records.

```
|where action in ('allow', 'accept')

|fields action, src, src_port , dest, dest_port, bytes_in, bytes_
out

|head 10
```

Example 2: Counts events per product, sorts by highest count, and returns only the top four products.

```
ocsf | aggr count as cnt by metadata.product.name | sort cnt desc
| head 4
```

help

Use the `help` operator to get documentation information about AQL operators and system lookup tables.

Syntax

`[|] help [help_type = {"operator" | "lookup"}]`

`[ | ]`	Optional. For this operator only, specifying the pipe is optional.
`[help_type = {"operator | lookup"}]`	Optional. Specify whether to return content for operators or system lookup tables. If the option is not specified, content for operators is returned.

Usage

You can use the `help` operator to return the set of AQL operator man pages as rows in the results table.

For operators, the columns are name, description, syntax, and examples.

For system lookup tables, the columns are name, description, and columns.

Examples

Example 1: Display help for operators.

```
|help
```

Example 2: Get documentation on specific operators.

```
|help help_type = "lookup"
```

Example 3: Returns help documentation for the lookup operator type within the AQL help system.

```
ocsf | help help_type='lookup'
```

history

Use the `history` operator to return the history of your most recent searches.

Syntax

```
|history [count=<num>]
```

`[count=<num>]`	Specify a limit to the number of searches returned.

Usage

You can use the `history` operator to return an in memory table, which displays your previous searches, sorted by the runtime and limited by the count field. The results table has the following fields: `event_time`, which in this case is the time the search was submitted; and `query`, which is the search expression that was submitted.

> **Note:** You must add the `anomali_api_proxy` keyword as the first keyword in your query.

By default, the `history` operator returns the history of the 100 most recent searches, up to a maximum of 500 searches.

Examples

Example 1: Retrieve historical API data.

```
anomali_api_proxy | history
```

Example 2: Query previous state information.

```
anomali_api_proxy | history count=10
```

Example 3: Access change history data.

```
anomali_api_proxy | history count=10 | fields query_last_run_time
as event_time, query
```

Example 4: Returns the 10 most recent search queries executed in the current session.

```
ocsf | history count=10
```

import_to_ts

Use the `import_to_ts` operator to send data from a specified field, together with the metadata specified as arguments, to ThreatStream.

Syntax

```
|import_to_ts <field> [ioc_confidence = <value>]
[ioc_classification = {"public"|"private"}] [ioc_
threat_type = <value>] [ioc_tags = <value>] [ioc_
public_tags = <value>] [override_system_confidence =
{true|false}] [auto_approve = {true|false}]
```

`<field>`	Event search field having data to be imported as observables into ThreatStream.
`[ioc_confidence = <value>]`	Confidence score integer value to assign to the batch of indicators. Default: `50`
`[ioc_classification = {"public"|"private"}]`	Classification string value to assign to the batch of indicators. Default: `private`
`[ioc_threat_type = <value>]`	Threat Type string value to assign to the batch of indicators. Default: `"malware"`
`[ioc_tags = <value>]`	Comma-separated string of private tags to assign to the batch of indicators. Default: no tags
`[ioc_public_tags = <value>]`	Comma-separated string of public tags to assign to the batch of indicators. Default: no tags
`[override_system_confidence = {true|false}]`	Boolean value to indicate whether the calculated confidence score should be completely overridden by the supplied confidence score. Default: `false`

`[auto_approve = {true|false}]`	Boolean value to indicate whether or not the import job is automatically approved or requires explicit approval. Default: `false` **Note**: Observables can only be auto approved when the observables are not public (in other words, when `ioc_classification = "private"`.)

Usage

You can use the `import_to_ts` operator to create ThreatStream import jobs based on the AQL query results. The operator options set metadata for the imported observable. If the operator options are not specified, the default values are applied to the imported observable.

The `import_to_ts` operator can be used to automate import jobs previously managed by analysts using the ThreatStream UI. For more information about importing ThreatStream observables, refer to Importing Observables in ThreatStream online help.

You can use the `import_to_ts` operator in conjunction with your organization's observables management practices and ThreatStream Integrator rules to orchestrate updates to Integrator destinations. For example, you can add tags to observables or set confidence scores for the observables so that the observables are included in updates to Integrator destinations. For information about ThreatStream Integrator, refer to ThreatStream online help.

Examples

Example 1: Import source IP addresses into ThreatStream.

```
|import_to_ts src_ip
```

Example 2: Import IP addresses with custom metadata settings.

```
|import_to_ts src_ip

ioc_classification="private"
```

```
ioc_confidence=80

ioc_threat_type="apt
```

inlist

Use the `inlist` operator to check if a field value is present in a lookup table column.

Syntax

```
|where <field_for_filtering_values> inlist (<lookup_table> <lookup_field>)
```

`<field_for_filtering_values>`	Specifies the name of the field to be used to filter the list of values.
`<lookup_table>`	Specifies the lookup table name which contains the field with the list of values to filter.
`<lookup_field>`	Specifies the name of the field in the `<lookup_table>` which contains the list of values to be included in the filter.

Usage

You can use the `inlist` operator to check field values in a list. This list of values comes from a lookup table, causing it to be modified dynamically.

Examples

Example 1: Filter results for source IP addresses present in a blacklist lookup table.

```
| where src_ip inlist (BlackListLookup ip)
```

Example 2: Filter results for users that match a contract employees lookup table.

```
| where user inlist (ContractEmployeesLookup user)
```

Example 4: Counts OCSF events per destination IP, excluding any IPs that appear in the `qa_tmp_table_aggr_test` lookup table.

```
ocsf | fields dst_endpoint.hostname |where dst_endpoint.hostname
inlist(ocsf_test_table dst_endpoint.hostname) | aggr count by
dst_endpoint.hostname
```

iplocation

Use the `iplocation` operator to enrich results with geolocation fields provided from a third-party database and associated with the IP address specified in the event log IP address field.

Syntax

```
|iplocation [prefix=<unique_geolocation_prefix>]
[allfields = [true, false]] <fields_included>
```

`<field>`	Specify an IP address field, such as `src_ip` or `dest_ip`.

`<unique_geolocation_prefix>`	Specify a prefix for geolocation fieldnames in results. For example, if you specify `geo_`, the column having the city in results is shown as `geo_city`. **Note:** If you require geolocation enrichment for both source and destination IPs, you must provide two unique geolocation prefixes, one for each instance of the `iplocation` operator. These prefixes prevent erroneous duplicate field names. See the "Examples" on the next page below to understand how to construct a query with multiple iplocation prefixes.
`allfields = [true, false]`	When `allfields=true`, results include `city`, `country`, `lat`, `lon`, `continent`, `postal_code`, `time_zone`. Otherwise, results include `city`, `country`, `lat`, and `lon`.

Usage

The `iplocation` operator can be used to create Geo Map visualizations.

Examples

Example 1: Enrich results with geolocation data for source IP addresses.

```
|iplocation src_ip
```

Example 2: Add geolocation fields with a custom prefix.

```
|iplocation prefix=geo_ src_ip
```

Example 3: Retrieve 90 days of login or session activity for a specific user, geolocate both source and destination IPs with prefixed fields to avoid collision, and get a clean activity timeline. Here, "`srcgeo_`" and "`destgeo_`" prefixes ensure

information about both source and destination cities, countries, and latitude-longitude is accurately captured.

```
| where user = 'john.doe@acme.com' and event_time &gt;= relative_
time(now(), -90d)

    and (action contains_ci 'login' or action contains_ci 'logon'

    or action contains_ci 'authenticate' or action contains_ci
'signin'

    or action contains_ci 'access' or action contains_ci
'session')

  | iplocation prefix=srcgeo_ allfields=true src_ip

  | iplocation prefix=destgeo_ allfields=true dest_ip

  | fields event_time, user, src_ip, dest_ip, host,

    srcgeo_city, srcgeo_country, srcgeo_lat, srcgeo_lon,

    destgeo_city, destgeo_country, destgeo_lat, destgeo_lon
```

Example 4: Include all available geolocation fields in the output.

```
|iplocation allfields=true src_ip
```

Example 5: Enrich non-empty source IPs with geolocation data, and counts events per IP grouped by country and coordinates.

```
ocsf | where src_endpoint.ip!='' | iplocation src_endpoint.ip |
aggr count as cnt by src_endpoint.ip, src_
endpoint.location.country, lat, lon
```

jira

Use the `jira` operator to create or update an investigation in Jira.

Syntax

```
|jira <field or ID> [project="<project name>"]
[issue="<ticket type>"]
```

`<field or ID>`	ThreatStream investigation ID field or ThreatStream investigation ID.
`<project name>`	Jira project name. Enclose the value in single or double quotes.
`<ticket type>`	Jira issue type. Enclose the value in single or double quotes.

Usage

You can use the `jira` operator to create or update an investigation in Jira, provided Jira has been integrated with your ThreatStream organization. The operator adds `JIRA ID` to results. If project and issue type are not specified, the update uses the default project and issue configured in the ThreatStream Jira integration settings.

Examples

Example 1: Create a Jira issue using an investigation ID without quotes.

```
|jira 123456
```

Example 2: Create a Jira issue using a single-quoted investigation ID.

```
|jira '123456'
```

Example 3: Create a Jira issue using a double-quoted investigation ID.

```
|jira "123456"
```

Example 4: Create a Jira issue from a calculated investigation ID field with specified project and issue type.

```
|calc ts_investigaion_id = 123456 |jira ts_investigaion_id
project='INTS' issue ='Task'
```

join

Use the `join` operator to join tables, based on either the eventlog or OCSF schemas. See "OCSF Schema Overview" on page 39 for more information.

The join operator also offers the `match` keyword to compare string fields in joins for wildcard-based matching when joining data with dynamic lookup tables. This enables flexible matching of string fields wherein `"*"` in the lookup table value matches any value in the join field, allowing you to filter different data combinations from search results.

Syntax

```
|join [join_type] ([your_subsearch] | <right_table>)
on <left_field>=<right_field> output <right_field>
[,<right_field>, ...]
```

With match keyword:

Single comparison: `join left <lookup_table> on <field1> match <field2>`

Multiple comparisons: `join left <lookup_table> on <field1> match <field2> and <field3> match <field4>`

`join [join_type] <table>`	`join_type`: inner (default), left, right, and full (or outer) `<table>` specifies the table to be joined with the event log table.

`<left_field>=<right_field>`	Specify the field used to join the event log and the other data source table. For example, if the event log field `src_ip` and the data source table field is `geoip`, specify the following: `src_ip=geo_ip`
`output right_field1, right_field2, rightfield3`	Specify the field names from the joined table to add to Event Search results. For example, if the joined table has fields `country`, `city`, and `zipcode`, specify the following: `output country, city, zipcode`
`[your_subsearch]`	The subsearch embedded within the primary search, inside square brackets (`[]`). See Subsearch documentation for more information.
`<field1> match <field2>`	`<field1>` is required. The event data field (left side of the join). `<field2>` is required. The lookup table field (right side of the join). This field can contain `"*"` as a wildcard for matching

`<field1> match <field2> and <field3> match <field4>`	Multiple comparisons, respectively, between • `<field1>` and `<field2>` • `<field3>` and `<field4>`

Usage

The `join` operator is used to join Event Search results with other data sources to correlate event log fields with enrichment fields that give you more insight into IT operations and security events.

You can use the match operator only for string fields on both sides. `"*"` in the lookup table acts as a wildcard and matches any value in the event field. The match operator also supports multiple comparisons, wherein each "*" acts as a wildcard for its respective field. Hence, you can have rows with one, several, or all fields set to "*", enabling highly flexible matching.

> **Note:** The match operator does not support wildcards for numeric fields. Hence, your lookup table must contain the exact numeric values.

Example

Example 1: Filters events to Palo Alto CEF logs, then enriches each event by joining against the `assetdb` lookup on source IP to append the asset country and hostname.

```
|where sourcetype contains "pan_cef" |join assetdb on src=ip
output ip, country, hostname as asset_host
```

Example 2: Filters to GCP logs, selects network flow fields, then left joins against `example_lookup_table` matching on destination IP and port to append the matched source IP as `final_output`.

```
|where sourcetype contains "gcp"

|fields sourcetype, src_ip, src_port, dest_ip, dest_port

|join left example_lookup_table on dest_ip match src_ip and dest_
port=dest_port output src_ip as final_output
```

Example 3:

> **Note:** Only the first 10000 result rows are considered as the output from a subsearch, which are then passed on to the primary search query. See Subsearch to understand how subsearches work.

```
example_table | calc Corr_id = json_extract (message, 'svr_corr_
id')

 | aggr latest(Corr_id) as correlation_id by svr_corr_id

 | where Hostname = "ASGREC05"

 | join left [Hostname_indexes | calc username = "johndoe" |
fields logs_types, Corr_id | where Hostname = "ASGREC05"] on
Hostname match Hostname output indexcount_as_host as indexcount_
as_host, Corr_id as correlation_id
```

Example 4: Counts events per source IP and retains only those enriched via lookup (matched IPs).

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

```
ocsf

| where src_endpoint.ip != ''
```

```
| aggr count as cnt by src_endpoint.ip
| lookup domain_ip on src_endpoint.ip=ip output ip
| where ip is not null
```

kmeans

Use the `kmeans` operator to perform kmeans clustering on a dataset containing numerical values to determine their cluster groupings based on the number of clusters required.

When using the elbow method, the operator directly displays the number of clusters, distortion, and the number of observations in each cluster.

> **Note:** Depending on your data type, consider using a scaling method such as the `StandardScaler` operator before using the `kmeans` operator.

Syntax

```
|kmeans <field-list> clusters=<clusters>
showcentroid=<showcentroid, bool> max_iter=<max_
iter>
```

`<field-list>`	A comma-separated list of fields to have kmeans clustering applied.

`clusters=<clusters>`	Specify the number of clusters the dataset is to be grouped into. Can either be an integer or an integer range. The format for integer range is <val1>-<val2>. For example: `|kmeans field1 clusters=<cluster_number1>-<cluster_number2>` When a range is applied, the output becomes be a table with the distortion and the number of elements in each cluster.
`showcentroid=<showcentroid, bool>`	Boolean field to determine if the centroid values are to be shown. The default is true. If false, the data points will only be grouped into the clusters without the centroids shown.
`max_iter=<max_iter>`	Maximum number of iterations the kmeans algorithm will be applied. The default is 10000.
`tol`	Convergence tolerance. The default if tol is not specified is 0.001.

Usage

The `kmeans` operator uses the kmeans machine learning algorithm to partition `n` observations into `k` clusters in which each observation belongs to the cluster with the nearest mean. When the `clusters` argument is specified as a range, this can instead be used to analyze the distortion of the cluster to determine the best number of clusters for the dataset.

Examples

To use kmeans with eventlog:

Example 1: Perform kmeans clustering on multiple fields with default cluster count.

```
|kmeans field1, field2. . . fieldn
```

Example 2: Perform kmeans clustering with a specific number of clusters.

```
|kmeans field1 . . . fieldn clusters=<cluster_number>
```

Example 3: Use the elbow method to determine optimal cluster count.

```
|kmeans field1 clusters=<cluster_number1>-<cluster_number2>
```

To cluster events by product name using the OCSF Schema:

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Example 4: Cluster OCSF events by product name using the K-means algorithm.

```
ocsf | aggr count as cnt b
```

like

Use the `like` expression to perform SQL-style pattern matching against string fields. It supports wildcard characters for flexible string comparisons and can be used anywhere a boolean expression is accepted.

Syntax

```
| like(<field_to_evaluate>, <pattern_for_matching>)
| <field_to_evaluate> like <pattern_for_matching>
```

`<field_to_evaluate>`	The string field to evaluate.
`<pattern_for_matching>`	The pattern to match against. Supports the wildcard characters described below. The pattern can be a string literal or a field reference.

Wildcards

%	Matches zero or more characters.
_	Matches exactly one character.

Usage

Use `like` anywhere a boolean expression is accepted, including `where` filters and conditional functions such as `if()`.

To negate a match, prefix the expression with `NOT`:

- `NOT like(<field>, <pattern>)`
- `<field> NOT like <pattern>`

The pattern argument can be a string literal or a field reference, allowing dynamic patterns constructed at query time using functions such as `replace()`.

This expression works with both `eventlog` and `ocsf` schemas. For OCSF fields, use dotted notation to reference nested fields.

Examples

Example 1: Eventlog pattern matching

The following example uses both the function and infix syntax to match usernames that start with `admin`. The second form also demonstrates a single-character wildcard to match any three-character value beginning and ending with `a`:

```
| where like(user, 'admin%')

| where user like 'a_a'
```

Example 2: OCSF with dotted notation

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

The following example filters OCSF events where a nested interface name field matches an exact value, then returns only that field in the results:

```
ocsf | where like(src_endpoint.interface_name, 'port1') | fields
src_endpoint.interface_name
```

Example 3: Inside a conditional expression

The following example uses `like` inside an `if()` function to label each event based on whether the username matches a pattern:

```
| calc result = if(like(user, 'admin%'), 'admin user', 'other
user')
```

Example 4: Dynamic pattern with replace()

The following example builds a `like` pattern dynamically by converting a glob-style wildcard (`*`) into the SQL-style wildcard (`%`) using the `replace()` function:

```
| calc dest_pattern = 'admin*' | where user like replace(dest_
pattern, '*', '%')
```

limit

Use the `limit` operator to specify a maximum number of rows returned.

Syntax

```
|limit <number>
```

Usage

Typically, use the `limit` operator together with the `sort` operator to filter results as you like—for example, to filter for the "top 10" results.

> **Note:** The maximum rows that can be returned to the Search results table is 100,000.

Examples

Example 1: Filters to allowed or accepted connections with known inbound byte counts, selects core network flow fields, and returns the top 100 heaviest inbound transfers.

```
| where action in ('allow', 'accept') and bytes_in is not null

| fields action, src, src_port, dest, dest_port, bytes_in, bytes_
out

| sort bytes_in desc

| limit 100
```

Example 2: Counts events per source hostname for non-empty products, sorts by highest count, and caps the output at 100 rows.

```
ocsf | where metadata.product.name!='' | aggr count as src_cnt by
src_endpoint.hostname | sort src_cnt desc | limit 100
```

listmodels

Use the `listmodels` operator to return a list of machine learning models that were learned using the `fit` operator.

Syntax

`|listmodels`

Usage

You can use the `listmodels` operator to return a list of machine learning models that were learned using the `fit` operator. The algorithm and arguments given when `fit` was invoked are displayed for each model.

Examples

Example 1: Return a list of all trained machine learning models.

```
|listmodels
```

listinv

Use the `listinv` operator to retrieve a list of ThreatStream investigations.

Syntax

`|listinv [inv_status="<status>"]`

`inv_status="<status>"`	Filter results by status. Specify `all`, `completed`, `pending`, `in-progress`, `unassigned`. If this option is not specified, all investigations are returned.

Usage

You can use the `listinv` operator to retrieve a list of ThreatStream investigations.

For information about ThreatStream investigations, refer to ThreatStream online help.

Examples

Example 1: Returns all investigations using the default status filter.

```
|listinv
```

Example 2: Returns all investigations regardless of status.

```
|listinv inv_status="all"
```

Example 3: Returns only investigations in a pending state.

```
|listinv inv_status="pending"
```

Example 4: Retrieves all investigations regardless of status, then counts them grouped by status to give a quick breakdown of open, closed, and other investigation statuses.

```
ocsf | listinv inv_status='all' | aggr count by status
```

listopinv

Use the `listopinv` operator to retrieve a list of open ThreatStream investigations.

Syntax

`|listopinv`

Usage

You can use the `listopinv` operator to retrieve a list of open ThreatStream investigations. This operator is similar to the `listinv` operator, which takes a `inv_status` argument to retrieve information about ThreatStream investigations having the specified status. The `listopinv` operator retrieves only the investigations having the status `open`.

For information about ThreatStream investigations, refer to ThreatStream online help.

> **Note:** The AQL query must specify a lookup table before this piped operator. It can be any lookup table.

Examples

Example 1: Queries open investigations sourced from the specified lookup table, returning all currently active investigation records associated with it.

```
<lookuptable>|listopinv
```

Example 2: Retrieves all open investigations and counts them by status.

```
ocsf | listopinv | aggr count as cnt by status
```

loadjob

Use the `loadjob` operator to load events or results of a previously completed search job query.

Syntax

`|loadjob job_id [ignore_running=bool]`

`<job_id>`	The search ID of the job whose artifacts need to be loaded. For example `68e22574dafb45ce92937f2094f2d21d`.
`<ignore_running>`	Optional. Skips over artifacts whose search is still running. By default, this is set to `true`.

Usage

The `loadjob` operator can be used for a variety of purposes. For example, you can issue a fairly expensive search and then calculate and display relevant statistics.

Examples

The following command will display results of job with id `68e22574dafb45ce92937f2094f2d21d`:

Example 1: Load results from a previously completed search job.

```
|loadjob 68e22574dafb45ce92937f2094f2d21d ignore_running=true
```

Controlling truncation in search

To improve the speed of searches, Security Analytics truncates the output of each search by default. For example, if the full output set is 10,000 events, the `loadjob` operator returns only a truncated portion of those results.

lookup

Use the `lookup` operator to query a source of content other than event logs.

Syntax

> **Note:** For the `insubnet` operator to function correctly, recreate your lookup table if you created it before January 2025. This is because in lookup tables created after Jan 2025, the lookup table automatically creates `<field_name>_network_start` and `<field_name>_network_end`, which are necessary for `insubnet` to return the correct output.

```
|lookup <lookup_table> on <condition1> [and
<condition2> ...] output <lookup_field> [, <lookup_
field>, ...]
```

`<lookup_table>`	Name of the lookup table that was set when the lookup table was uploaded with the API.

`<condition1> [and <condition2> ...]`	**Note:** For the `insubnet` operator to function correctly, recreate your lookup table if you created it before January 2025. This is because in lookup tables created after Jan 2025, the lookup table automatically creates `<field_name>_network_start` and `<field_name>_network_end`, which are necessary for `insubnet` to return the correct output.
	
	Specify a lookup condition using the following condition types:
	
	• `<event_field>=<lookup_field>`: Fields that join the event log and the lookup table.
	
	For example, if the event log field `src_ip` and the lookup table field is `geoip`, specify the following:
	
	`src_ip=geo_ip`
	
	• `<event_field> insubnet <lookup_field>`: Returns TRUE if the `<event_field>` falls within the subnet in the `<lookup_field>`; otherwise, returns FALSE.
	
	For example, to check if `src_ip` IP address falls within the `src_ip_cidr` subnet, specify the following:
	
	`src_ip insubnet src_ip_cidr`
	
	• `<event_field> like <lookup_field>`: Evaluate whether the `<event_field>` matches the pattern specified in the `<lookup_field>`. Specify the pattern using the `calc` operator.
	
	For example:
	
	`|calc url = if(url='', "https://www.salesmanagopush.com/index.html", url)`
	
	`|lookup example_lookup_table on url like url`

	You can specify multiple conditions using the AND operator. Conditions can be of any type specified above. For example, `url like url and dest like dest and src_ip insubnet src_ip_cidr`
`output <lookup_field1>, <lookup_field2>`	Specify the fieldnames from the lookup table to add to Event Search results. For example, if the lookup table has fields `country`, `city`, and `zipcode`, specify the following: `output country, city, zipcode`

Usage

A *lookup table* is a CSV file that has enrichment data—for example, the commonly used MaxMind IP geolocation data. A lookup table can be any set of fields that you want to correlate with an event log field, enabling the lookup table data to enrich Event Search results.

See Managing Lookup Tables for information about uploading a lookup table.

The `lookup` operator is used to join Event Search results with the specified lookup table to correlate event log fields with enrichment fields that give you more insight into IT operations and security events.

> **Tip:** In the Event Search text search box, enter the lookup table name with no other syntax to display the contents of the lookup table. For example, to display the contents of a lookup table named geoinfo, type `geoinfo` in the Event Search text box and click **Search**.

Examples

Example 1: Join event results with geolocation lookup data using a field match.

```
geoinfo

|where sourcetype contains "pan_cef" |lookup geoinfo on src=ip
output ip, country, hostname as asset_host |where country = "US"
```

Example 2: Join event results using subnet and pattern matching conditions.

> **Note:** For the `insubnet` operator to function correctly, recreate your lookup table if you created it before January 2025. This is because in lookup tables created after Jan 2025, the lookup table automatically creates `<field_name>_network_start` and `<field_name>_network_end`, which are necessary for `insubnet` to return the correct output.

```
example_lookup_table

|where sourcetype="ms_defender" and src_ip != "" and not src_ip
contains ":"

|calc src_ip = if(src_ip startswith '54.240.' , '0.0.0.0', src_
ip)

|calc dest = if(isnull(dest), "www.salesmanagopush.com", dest)

|calc url = if(url='',
"https://www.salesmanagopush.com/index.html", url)

|lookup example_lookup_table on src_ip insubnet src_ip_cidr and
dest like dest output src_ip_cidr, url as up_url
```

In the example above, the `insubnet` will return the correct output if the `example_lookup_table` lookup table was created after January 2025.

Example 3: Filters OCSF events to those with both source and destination IPs, then enriches each event by looking up the source IP against the `domain_ip` lookup table to append country, IP, and hostname details.

```
ocsf | where src_endpoint.ip!='' and dst_endpoint.ip!='' | lookup
domain_ip on src_endpoint.ip=ip output country, ip, hostname
```

lstintdest

Use the `lstintdest` operator to list the configured Integrator destinations by integrator instance.

Syntax

`|lstintdest`

Usage

You can use the `lstintdest` operator to list the names of the Integrator instances and the configured Integrator destinations, using the following columns:

- `integrator_name`: names of the Integrator instances
- `configured_destinations`: configured Integrator destinations

For information about ThreatStream Integrator, refer to ThreatStream online help.

Examples

Example 1: List all Integrator instances and their configured destinations.

```
|lstintdest
```

Example 2: Returns a list of all available integration destinations configured in the system, adhering to the OCSF schema.

```
ocsf |lstintdest
```

lstperftag

Use the `lstperftag` operator to return your organization's ThreatStream Preferred Tags.

Syntax

`|lstperftag`

Usage

You can use the `lstperftag` operator to return your organization's ThreatStream Preferred Tags. Two columns are added to results: `tag` and `created by`.

For information about ThreatStream observable tags, refer to ThreatStream online help.

Example

Example 1: Return all Preferred Tags configured for the organization.

```
|lstperftag
```

Example 2: Returns a list of all available performance tags defined in the system.

```
ocsf | lstperftag
```

makejson

Use the `makejson` operator to create a JSON object from the specified set of fields in the search results and place the JSON into a new field.

Syntax

`|makejson [fields] output=<string>`

`[fields]`	Optional list of fields to include in the resulting JSON. If no fields are specified, then all available fields are included.
`output=<string>`	Specify the name of a new field to contain the JSON object.

Usage

You can use the `makejson` operator to convert results into a JSON object for further processing.

Examples

Example 1: Create a JSON object from selected fields.

```
|fields sourcetype, app_type |makejson sourcetype, app_type
output=new_json
```

Example 2: Create a JSON object from selected fields and sort by ID.

```
|makejson_tests |fields id, products, net_income |makejson id,
products, net_income output=new_field |sort id asc
```

Example 3: Create a JSON object from multiple fields with complex sorting.

```
|makejson_tests |fields id, products, quarter, net_income,
reached |makejson id, products, quarter, net_income, reached
output=new_field |sort new_field asc
```

makemv

Use the `makemv` operator to parse a specified field into a multivalue field having a list of values produced by the parsing function.

Syntax

```
|makemv [delim="<chars>"| tokenizer=<regex>] [allowempty =
{true|false}] [setsv={true|false}] <field>
```

`delim="<char>"`	Specify a single character or multiple characters present in the specified field to use as the delimiter for creating the multivalue field. If this option is not specified, the implicit default is a single space (" ").
`tokenizer=<regex>`	Specify a regular expression with a capturing group that is repeat-matched against the values in the field. For each match, the first capturing group is used as a value in the newly created multivalue field.

`[allowempty = {true|false}]`	Specify whether to permit empty string values in the multivalue field. When using `allowempty=true`, repeats of the delimiter string produce empty string values in the multivalue field. If this option is not specified, the implicit default is `allowempty=false`. For example, if `delim=","` and `field="a,,b"`, by default, the operator does not produce any value for the empty string. When using the tokenizer argument, zero length matches produce empty string values. By default they produce no values.
`[setsv= {true|false}]`	Specify whether to replace the value of the specified input field with the results of parsing by this operator. When `setsv=true`, `makemv` combines the decided values of the field into a single value, which is set on the same field. (The simultaneous existence of a multivalue and a single value for the same field is a problematic aspect of this flag.) If this option is not specified, the implicit default is `setsv=false`.
`<field>`	Specify the field with the multivalue content to be parsed by this function.

Usage

You can use the `makemv` operator when you want to make use of the product of the parsing function. For example, you might want to perform filtering or sorting based on part of an IP address or URL.

Examples

Example 1: Parse a field into a multivalue field using space delimiter.

```
|fields sourcetype |makemv delim=" " sourcetype
```

Example 2: Parse an IP address field into separate octets using dot delimiter.

```
|fields event_time, src |makemv delim="." src |mvexpand src_mv
```

Example 3: Parse an IP address field with specified source IP filtering.

```
|where src_ip = "172.18.26.2"| fields event_time, src_ip | makemv
del
```

Example 4: Takes the first 10 non-null source IPs, splits each IP address into a multi-value field using . as the delimiter, then reverses the order of the resulting values (for example, turning 192.168.1.1 into [1, 1, 168, 192]).

```
ocsf | fields src_endpoint.ip | where src_endpoint.ip is not
null| limit 10 | makemv delim = '.' src_endpoint.ip | mvreverse
(src_endpoint.ip_mv)
```

makeresults

Use the `makeresults` operator to generate a temporary table, which can then be used in subsequent searches.

Syntax

With the pipe (|) symbol: `| makeresults format={csv|json} input={<csv>|<json>} [count=<int>]`

Without the pipe symbol: `makeresults format={csv|json} input={<csv>|<json>} [count=<int>]`

`format={csv|json}`	Specify the format of your input string. If format is not specified, the operator cannot take an input string. Instead, it generates a "dummy" table with a column having the event_time.

`input={<csv>|<json>}`	If format is CSV, specify a string similar to the following:
	`input="val1,val2\n1,2\n3,4\n,5\n6"`
	In this example, the `\n` is a newlilne character, so the string returns a table with the following rows:
	• header row: (event_time, val1, val2)
	• row 1: (current_time, 1,2)
	• row 2: (current_time, 3,4)
	• row 3: (current_time, 5,6)
	If format is JSON, specify a string similar to:
	`input='[{"val1": "test1"},{"val2": "test2"}]'`
	The example string returns a table with the following rows:
	• header row: (event_time, _raw, val1, val2)
	• row 1: (current_time, {"val1": "test1"}, "test1", <empty cell>)
	• row 2: (current_time, {"val2": "test2"}, <empty cell>, "test2")

`[count=<int>]`	When no input string is specified, specify a count. When there is no input, the operator creates a table having the field event_time of length count.

Usage

You can use the `makeresults` operator to make custom search tables on the fly using either comma-separated value (CSV) input (comma and newline separated) or JSON input (shallow key search to generate table names).

You can also use the `makeresults` operator to create "dummy" tables to be reused for future operator uses. The dummy tables have one column having the generated event time.

Examples

> **Note:** For CSV formats, you must place the input in double quotes. Not doing so will treat `\n` as a string and not the newline character.

Example 1: Creates a single-column inline result set with two distinct values (test and test2) injected directly into the pipeline.

```
makeresults format='csv' input="test\ntest\ntest2"
```

Example 2: Creates a single-row inline result set from a JSON array with one field-value pair, injecting it directly into the pipeline.

```
| makeresults format="json" input='[{"test": "test"}]'
```

Example 3: Creates an inline two-row result set from a JSON array with two fields, injecting structured test data directly into the pipeline without querying any data source.

```
ocsf | makeresults format='json' input='[{"test1": "valu1test1",
"test2": "value1test2"},{"test1": "valu2test1", "test2":
"value2test2"}]'
```

mcatalog

Use the `mcatalog` operator to generate a list of all distinct values for a field.

Syntax

```
|mcatalog values(<field>) [where <boolean-
expression>] [by <field-list>]
```

`values(<field>)`	Specify the field for which to generate the list of distinct values.

`[where <boolean-expression>]`	Optional. Specify a filter condition. The filter supports the same boolean expressions as the "where" on page 700 operator.
`[by <field-list>]`	Optional. Specify a list of fields to group the results.

Usage

You can use the `mcatalog` operator when there is a need to find all the distinct values in a field.

You can also use `aggr distinct()` to find distinct values. The `mcatalog` operator is the better choice when you also want to use the `where` option to apply a filter condition on the list of values.

Examples

Example 1: Generate a list of distinct city values.

```
|mcatalog values(city)
```

Example 2: Generate distinct city values grouped by category and sorted.

```
|mcatalog values(city) by category | sort category asc
```

Example 3: Generate distinct city values for a specific category.

```
|mcatalog values(city) where category = 'Furniture' by category
```

Example 4: Generate distinct status values for multiple filtering conditions grouped by location.

```
|mcatalog values(status) where country = 'United States' and
category = 'Technology' by city, sub_categ
```

mdti_ssl

> **Note:** `mdti_ssl` operator was formerly called the `riskiq_ssl` operator.

Use the `mdti_ssl` operator to enrich results related to an IP address or domain name or field containing an IP address or domain name using the ThreatStream MDTI enrichment.

Syntax

```
|mdti_ssl <field or value>
```

`<field or value>`	Specify the field name or IP address or domain name to look up.

Usage

You can use the `mdti_ssl` operator to enrich results with data from the ThreatStream MDTI integration.

Examples

Example 1: Enrich source IP addresses with MDTI enrichment data.

```
|mdti_ssl src_ip
```

Example 2: Enrich a specific domain with MDTI enrichment data.

```
|mdti_ssl 'example.com'
```

Example 3: Enrich a specific IP address with MDTI enrichment data.

```
|mdti_ssl '192.0.2.1'
```

Example 4: Queries Microsoft Defender Threat Intelligence for SSL certificate data associated with the IP 91.92.253.35

```
ocsf | mdti_ssl '91.92.253.35'
```

metasearch

Use the `metasearch` operator to get the list of tables existing in the database, including the `event_log` table and lookup tables.

Syntax

`|metasearch`

Usage

You can use the `metasearch` operator to get the list of tables existing in the database, including the `event_log` table and lookup tables. Only active tables are included. The query returns a single column named `table_name`.

Examples

Example 1: List all tables available in the database.

```
|metasearch
```

ml_model_summary

Use the `ml_model_summary` operator to retrieve the metadata about a machine learning model that was previously trained and saved using the `fit` operator, presenting a clear, more interpretable summary of the model parameters, including the model features and their associated coefficients.

Specifically, the summary now displays each feature alongside its coefficient, standard error, and p-value (where applicable) in a structured table.

This summary, therefore, helps users in critical model analysis and understanding of the model output and effectiveness.

Syntax

```
|ml_model_summary <model_name> [display_as_table]
```

`<model_name>`	Name of the model previously trained and saved using the `fit` operator. Model name is case sensitive.
`display_as_table`	Choose whether to display the summary as a table. Possible values are `true` or `false`.

Usage

Use the `ml_model_summary` operator to to retrieve the metadata about the machine learning model which has been trained and saved using the `fit` operator. The metadata it retrieves are the model algorithm, model parameters, and the model summary. The model summary fields vary for each algorithm. It might include model coefficients, intercepts, offsets, cluster details, and so on, depending on the model.

The summary table displays each feature alongside its coefficient, standard error, and p-value (where applicable), enabling you to:

- Identify the influence of each feature on the model outcome.
- Conduct statistical analysis or adjustments through p-values and standard errors. For example, you can:
  - Remove insignificant features.
    For example, if a feature has a high p-value (for example, greater than 0.5), its contribution might be insignificant.
  - Check for multicollinearity.
    Consider removing or combining features that are highly correlated.
  - Refit the model.
    If you remove or adjust some features, consider refitting the model.
- View results in a well-formatted table, simplifying model analysis.

Examples

Example 1: Retrieve model summary displayed as a formatted table.

```
|ml_model_summary my_model display_as_table=Tru
```

multikv

Use the `multikv` operator to parse tabular fields.

Syntax

```
|multikv <field> <column_names ...> [FILTER <filter_terms ...>]
[<option_name>=<option_value>] ...
```

`<field>`	Specify the field that contains the tabular data to be parsed.
`<column_names ...>`	Specify a space-separated list of column names to be extracted from the tabular data.
`[FILTER <filter_terms ...>]`	Optional. Specify a space-separated list of values to filter. If specified, only rows that contain at least one of the filter values are returned.
`[<option_name>=<option_value>] ...`	Optional. Specify one or more of the following options: • `copyattrs`: If set to `true`, columns from the original row are copied to the new ones created from it. If set to `false`, columns from the original row are not copied. • `forceheader`: Specified the line number of the table header. Count is 1-based. • `noheader`: Specify `true` if the table has no header. • `rmorig`: If set to `true`, original rows are excluded from the output. Setting it to `false` includes them. If this option is not specified, the implicit default is `true`. If set to `false`, set the `copyattrs` option to `true` as well. • `separator`: A designated single character or regular expression employed to delineate columns. If this option is not specified, the implicit default is one or more spaces.

Usage

You can use the `multikv` operator when you want to make use of the product of the parsing function. When applied, the operator extracts the specified fields into new columns, transforming each row of the table found within a single entry into its own separate entry. This means if an original entry contains a table with 5 rows and 5 columns, the `multikv` operator parses this into 5 new entries, each having the specified 5 columns extracted.

Examples

Example 1: Parse tabular data and extract multiple column values.

```
|multikv multiline_string NAME AGE OCCUPATION
```

Example 2: Parse tabular data without copying original row attributes.

```
|multikv multiline_string NAME OCCUPATION copyattrs=false
```

Example 3: Parse tabular data without copying attributes and exclude original rows.

```
|multikv multiline_string NAME OCCUPATION copyattrs=false
```

Example 4: Parse tabular data with filtering by specified values and include original rows.

```
|multikv multiline_string NAME AGE OCCUPATION filter Red Blue
rmorig=false
```

Example 5: Parse tabular data using custom column separator.

```
|multi
```

mvcombine

Use the `mvcombine` operator to reduce result rows by combining event logs that are identical except for values in the specified field list.

Syntax

```
|mvcombine [delim="<char>"] <field>
```

`[delim=<char>]`	Specifies the delimiter you want to see in the cell that has the combined result values—for example, a comma or a pipe.
`<field>`	Specifies the field for which to combine results. In the results table, the results for this field are displayed as an array having values separated by a delimiter.

Usage

You can use the `mvcombine` operator when you want to see the various values for a field in a single cell in the results table.

Examples

Example 1: Combine multiple action values into a single comma-separated field.

```
|where action != "" |fields sourcetype, action |head 100
|mvcombine delim="," action
```

Example 2: Combines all `activity_name` values for each unique `src_endpoint.hostname` into a single comma-delimited multi-value field, collapsing multiple rows per host into one.

```
ocsf | fields src_endpoint.hostname,activity_name | mvcombine
delim=',' activity_name
```

mvexpand

Use the `mvexpand` operator to rewrite a single row having a multivalue field to multiple rows—a row for each value formerly in the multivalue field.

Syntax

```
|mvexpand <field>
```

Usage

You can use the `mvexpand` operator when you want to distinguish individual values in the event having the multivalue field.

Examples

Example 1: Expand tags into separate rows and filter by tag value.

```
observables | calc tags_array = array_join(tags, ',') | mvexpand
tags | where tags_expanded contains_ci 'malware'
```

Example 2: Expand sourcetype field parsed by space delimiter into separate rows.

```
|fields sourcetype |makemv delim=" " sourcetype | mvexpand
sourcetype_mv
```

Example 3: Expand IP address field parsed by dot delimiter into separate rows.

```
|fields event_time, src |makemv delim="." src | mvexpand src_mv
```

Example 4: Expand source IP address field for specific IP with delimiter parsing.

```
|where src = "192.168.2.14" |fields event_time, src | makemv
delim "." src | mvexpand src_mv
```

Example 5: Takes two OCSF events, creates a comma-delimited test field, splits it into a multi-value field, reverses it, expands each value into its own row, converts the multi-value field back to a single value, then returns the expanded and reversed string values

```
ocsf | limit 2 | calc r.test='1,2' | fields r.test | makemv
delim=',' r.test | mvreverse(r.test) | mvexpand r.test_mv | nomv
r.test_mv --| return $r.test_mv_mv_str, r.test_reverse
```

mvreverse

Use the `mvreverse` operator to reverse the order of the values in a multivalue field.

Syntax

`|mvreverse(<field>)`

`<field>`	Specify a field name having multivalue data.

Usage

You can use the `mvreverse` operator with the calc operator `mvappend` function.

Examples

Example 1: Reverse the order of numeric array values parsed by dot delimiter.

```
|makemv delim = '.' num_array |mvreverse(num_array_mv)
```

Example 2: Reverse the order of string array values parsed by pipe delimiter.

```
|makemv delim = '|' str_array |mvreverse(str_array_mv)
```

Example 3: Reverse the order of a dynamically created multivalue array.

```
|calc a = mvappend(1,2,3) |mvreverse(a)
```

Example 4: Takes two OCSF events, creates a comma-delimited test field, splits it into a multi-value field, reverses it, expands each value into its own row, converts the multi-value field back to a single value, then returns the expanded and reversed string values

```
ocsf | limit 2 | calc r.test='1,2' | fields r.test | makemv
delim=',' r.test | mvreverse(r.test) | mvexpand r.test_mv | nomv
r.test_mv --| return $r.test_mv_mv_str, r.test_reverse
```

nomv

Use the `nomv` operator to separate values of a multivalue field with a newline character in the results table.

Syntax

`|nomv <field>`

Usage

You can use the `nomv` operator to improve readability of multivalue fields in results.

Examples

Example 1: Display aggregated actions separated by newlines for each sourcetype.

```
|where action != "" |head 1000 |aggr array_agg(action) as actions
by sourcetype |nomv actions
```

Example 2: Takes two OCSF events, creates a comma-delimited test field, splits it into a multi-value field, reverses it, expands each value into its own row, converts the multi-value field back to a single value, then returns the expanded and reversed string values

```
ocsf | limit 2 | calc r.test='1,2' | fields r.test | makemv
delim=',' r.test | mvreverse(r.test) | mvexpand r.test_mv | nomv
r.test_mv --| return $r.test_mv_mv_str, r.test_reverse
```

openports

Use the `openports` operator to enrich results related to an IP address or field containing an IP address using the ThreatStream OpenPorts enrichment.

Syntax

`|openports <field or value>`

`<field or value>`	Specify the field name or IP address to look up.

Usage

You can use the `openports` operator to enrich results that inform your security orchestration, automation and response actions. For each IP address passed to the enrichment, a list of open ports reported within the previous seven days is returned.

Examples

Example 1: Enrich source IP address field with open ports data.

```
|openports src_ip
```

Example 2: Enrich a specific IP address with open ports data.

```
|openports "192.0.2.1"
```

outlier

Use the `outlier` operator to modify numeric outputs for given input fields that contain outliers.

Syntax

```
|outlier [field_list=<field-list>] [outlier_action=
{"transform|remove"}] [mark={True|False}]
[param=<number>] [uselower={True|False}]
```

`field_list=<field-list>`	Optional. Specify a comma-separated or space-separated list of numeric fields that you need to modify.

`outlier_action={"transform|remove"}`
or
`outlier_action={"tf|rm"}`
	Optional. Possible options:
	• `"transform"`, or `"tf"`: Applies a transform to bound the outlier values.
	• `"remove"`, or `"rm"`: Removes all fields outside the threshold.
	Default: `"transform"`

`mark={True|False}`	Optional. Specify a boolean value to add a suffix of 000 to the output of a transformed outlier. Default: `False` **Note**: The output field will be a string type rather than a numeric type, so avoid using this field in graphical representations or further processing.
`param=<number>`	Optional. Specify a numeric value for calculating the outlier. The outlier is calculated as follows: `upper quartile + inter quartile range * param`. Default: `2.5` **Note**: The inter-quartile range is calculated from (upper quartile - lower quartile). If `uselower` is set to `True`, a similar calculation is used, but is as follows: `lower quartile - inter-quartile range * param`.
`uselower={True|False}`	Optional. Specify a boolean value to determine whether the outlier logic should be applied to outlier values below the mean. Default: `False`

Usage

You can use the `outlier` operator to transform or remove numeric values in specific fields. If you use the `outlier` operator without any attributes, it will try to transform all numeric fields.

The `outlier` operator processes datasets by filtering based on the inter-quartile range (IQR). The IQR is calculated as the difference between the 25th and 75th percentiles of numeric fields. Let's assume that a field's value in an event is less than `25th percentile - param*IQR` or greater than `75th percentile + param*IQR`. In that case, depending on the specified action parameter, either the field is transformed, or the event is removed.

Use `anomalydetection` to find fields that contain outliers.

Examples

Example 1: Transform all numeric fields to handle outliers with default parameters.

```
|outlier
```

Example 2: Transform a field to handle outliers below the mean.

```
|outlier val2 uselower=true
```

Example 3: Transform a field and mark outlier values with a suffix.

```
|outlier val2 mark=true
```

Example 4: Transform multiple fields to handle lower outliers.

```
|outlier val1 val2 uselower=true
```

Example 5: Remove outliers from results.

```
|outlier val1 action=remove
```

Example 6: Counts events by activity type, then identifies statistical outliers in those counts, marking and transforming anomalous activity volumes.

```
ocsf | aggr count as test by activity_name | outlier test
outlier_action='transform' mark=true
```

outputtext

Use the `outputtext` operator to return the contents of the `_raw` field as a new field named `xml` in results.

Syntax

`|outputtext [usexml={true|false}]`

`[usexml={true|false}]`	Optional. If set to `true`, the copy of the `_raw` field in the XML is escaped XML. The implicit setting if the option is not specified is `false`, meaning the XML field is an exact copy of the `_raw` field.

Usage

You can use the `outputtext` operator when you want to use the text from `_raw` in XML format—for example, to prepare reports.

Examples

Example 1: Output raw field contents as unescaped XML.

```
... |outputtext
```

predict

Use the `predict` operator to make predictions for time series data. The operator will generate predictions for in-sample data and forecasts for out-sample data.

Syntax

```
|predict <field>+ <model> [forecast_k=<int> | conf_
interval=<int> | holdback=<int> | trend=<bool> |
seasonal=<bool|string> | correlate=<field>] [with
<timestamp_field>]
```

`<field>+`	Required. Specify a comma-separated or space-separated list of numeric fields to predict.

`<model>`	Required. Specify the model to be used for prediction. Possible options: • `autoregressive`: It is an autoregressive AR-X (p) model using conditional maximum likelihood (OLS). When predicting using the autoregressive model, the number of lags to be included in the model needs to be calculated first (this is an automatic process performed by the operator). For example, depending on the data, the best lags can be included in the model: [1,4,12]. It means, data points: 1, 4, and 12 will be included in the model, therefore, data points from 1 to 12 will be missing for the predicted data. In this example, the calculated AR-X model is AR-X(12). AR-X(p) models can be up to AR-X(15), which means you need to provide at least ~30 data points to get results. If you want to get more accurate results, then you need to provide more data points. • `sarimax`: It is an Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors (SARIMAX) model. Similar to the autoregressive model, it can incorporate trend and seasonal variations when predicting. This model automatically selects seasonal orders based on data without the need to manually configure it. It is very powerful as combined with exogenous data can improve the predictive power of the model, therefore, improve the forecast accuracy.

`forecast_k=<int>`	Optional. Specify the number of points to be forecast. These points are beyond the data provided, therefore, if you want to forecast 12 points in a monthly data set, you can specify `forecast_k=12`. If you only want to generate a prediction for in-sample data, then you can specify `forecast_k=0`. Default: `5`
`conf_interval=<int>`	Optional. Specify the confidence interval for the forecasts made by the `predict` operator. It is a percentage representing the probability that the future values will fall within the specified range around the predicted values. Default: `95`
`holdback=<int>`	Optional. Specify a portion of data that should be withheld from the model training process. The data can be used to evaluate the model's performance. If you have 100 data points and set holdback to `10`, the last 10 data points will not be used in training the model. This applies only for the Autoregressive model. It will be ignored for the SARIMAX model. Default: `0`
`correlate=<field>`	Optional. Specify the field that can be used as exogenous data to be associated to the training model process. It must have the same number of observations as the list of fields to predict, otherwise, the training process will fail.
`trend=<bool>`	Optional. Specify whether you want to include a trend in your prediction. • `false`: No trend. • `true`: Constant and time trend. Default: `false`

`seasonal=<bool|string>`	Optional. • For the autoregressive model, it is a boolean value. Specify `true` if you want to include seasonal dummies in the model. Default: `false` • For the SARIMAX model, it must be a string containing 4 digits: `(P,D,Q,s)`. ▪ `D` indicates the integration order of the process. ▪ `P` and `Q` indicate the autoregressive (AR) and moving average (MA) orders. ▪ `s` is an integer indicating the periodicity (number of periods in a season). Default: `'0,0,0,0'`
`with <timestamp_field>`	Optional. Specify the time field to be used in the forecast. According to the last value of this field, the forecasts will be generated using the `forecast_k` parameter. It should be in the timestamp format in milliseconds.

Usage

You can use the `predict` operator to predict future behavioral data using different algorithms which can give different results. This way, you can easily find the best algorithm for the data.

> **Note:** To be able to get results, you need to provide time series data with only one frequency, otherwise, the operator will not be able to determine the frequency to use (if there are two or more) and will not generate results.

The `predict` operator can be an invaluable tool for forecasting and mitigating cybersecurity threats. It can be very useful when trying to forecast future time series data, or even to validate the model by comparing the predicted values with the actual values.

For each predicted field, three additional fields will be included in the results:

- `<field>_predicted`: Predicted point for the specific field.
- `<field>_lower`: Lower limit for the confidence interval for the predicted point.
- `<field>_upper`: Upper limit for the confidence interval for the predicted point.

Examples

Example 1: Predict cars field using autoregressive model with confidence interval and holdback.

```
test_predict_table | predict cars autoregressive conf_interval=95
forecast_k=5 holdback=5 seasonal=false with time_stamp | sort
time_stamp asc
```

Example 2: Predict field using SARIMAX model.

```
test_predict_table | predict cars sarimax conf_interval=95
forecast_k=5 seasonal='1,1,1,12' with time_stamp | sort time_
stamp asc
```

Example 3: Generates hourly event count predictions using an autoregressive model trained on the OCSF event time trend.

```
ocsf | timetrend count as cnt span=1h | predict cnt
autoregressive
```

rangemap

Use the `rangemap` operator to categorize the values returned for a specified numeric field.

Syntax

```
|rangemap field=<field_name> [<attribute>=<num>-
<num>,...] [default=<default_set>]
```

`field=<field_name>`	Specify a numeric field.
`[<attribute>=<num>-<num>]`	Optional. Specify a comma-separated list of attribute name-value pairs so that the attribute name is output when the attribute value lies within the numeric range.
`[default=<default_set>]`	Optional. Specify a default value to output when the input field does not match a range.

Usage

You can use the `rangemap` operator when you want to categorize results belonging to the specified ranges.

Examples

Example 1: Categorize sales values into ranges with color labels.

```
rangemap_operator_e2e_test |rangemap field=sales
attribute="red=0-1000,yellow=1001-3000,green=3001-6000"
default="gray"
```

Example 2: Categorize sales field with overlapping ranges and default value.

```
rangemap_operator_e2e_test |rangemap field=sales
attribute="red=1000-2500,yellow=1000-3000,green=3000-5000,blue=0-
10000" default="gray"
```

Example 3: Counts events per product, assigns a color label (red, yellow, or green) based on event volume ranges, defaults to gray for any count outside the defined ranges, then sorts by count.

```
ocsf | aggr count as type_count by metadata.product.name |
rangemap field= type_count attribute='red=0-100,yellow=101-
500,green=501-20000' default='gray' | sort type_count
```

rare

Use the `rare` operator to display the least common values for the specified field.

Syntax

```
rare [<number>] [options>...] <field>
```

Option	Usage
`countfield="<new field>"`	Create a new field and populate with a count of rare results.

`limit = <number>`	Specify a maximum number of rows. Same as specifying a number after rare.
	For example, the following are equivalent:
	`rare 2`
	`rare limit = 2`
	If no limit is specified, the default is 10.

`percentfield="<new field>"`	Create a new field and populate with a percentage of rare results.
`showcount= [true,false]`	If true, include a column in results showing the count of results having the rare values.
`showperc= [true,false]`	If true, include a column in results showing the percent of results having the rare values.

Examples

Example 1: Find rarest action values and count occurrences.

```
|where action != "" |rare countfield="action_count" act
```

Example 2: Returns the 11 rarest combinations of `activity_name`, `src_endpoint.hostname`, and `dst_endpoint.hostname`, suppressing the count column but showing the percentage of each combination of the total events in a pct field.

```
ocsf | rare 11 activity_name showcount=false src_
endpoint.hostname percentfield = 'pct' dst_endpoint.hostname
```

recordedfuture

Use the `recordedfuture` operator to enrich results with information from the ThreatStream RecordedFuture enrichment.

Syntax

`|recordedfuture <field>`

`<field>`	Specify a field name. Fields having IPv4 addresses, domain names, for file hashes can be used to pull in data from RecordedFuture.

Usage

You can use the `recordedfuture` operator to enrich results with information from the ThreatStream RecordedFuture enrichment.

Examples

Example 1: Enrich source IP address with RecordedFuture threat intelligence.

```
|recordedfuture src_ip
```

Example 2: Creates an inline list of two IPs and enriches each with Recorded Future threat intelligence data.

```
ocsf | makeresults format='csv'
input='ip\\n185.220.100.240\\n80.66.65.177' | recordedfuture ip
```

reltime

Use the `reltime` operator to calculate the time since the specified timestamp field and add a column having this relative time field to results.

Syntax

`|reltime timefield=<field> [prefix=<string>]`

`[timefield=<field>]`	Specify a timestamp field to use for the calculation.
`[prefix=<string>]`	Optional. Specify a prefix for the calculated field. If you do not specify the `prefix` argument, the new field name is `reltime`.

Usage

You can use the `reltime` operator to calculate how long since the time specified by the timestamp field. The operator creates a new field having a human readable relative time following the form `15 minutes ago`.

Examples

Example 1: Calculate relative time from earliest time timestamp field.

```
... |reltime timefield=earliest_time
```

Example 2: Demonstrates a comprehensive chain of time manipulation functions on a single OCSF event: converting Unix timestamps to human-readable format and back, computing relative time offsets, formatting dates, and parsing string timestamps.

```
ocsf

| limit 1 | fields event_time

| calc event_time=from_unixtime(event_time)

| calc unixtime=to_unixtime(event_time)
```

```
| calc tnow=now() | reltime timefield=unixtime

| calc relative_time_minus_1_day=relative_time(unixtime, -1d)

| calc ctime_field = ctime(relative_time_minus_1_day)

| calc mktime_field = mktime(event_time)

| calc formatted_time=date_format(event_time, '%Y-%m-%d')

| calc strp_time=strptime('2024-05-21 08:13:24', '%Y-%m-%d
%H:%M:%S')

| calc ttime_format=timeformat('2024-05-21 08:13:24', '%Y-%m-%d
%H:%M:%S')

| calc unixtime2=to_unixtime(ttime_format)
```

replace

Use the `replace` operator to change a particular value for a specified field in results returned. You can also use `regex_replace` as an alias for the same operator.

Syntax

```
replace <string> WITH <string> IN <field>
regex_replace <string> WITH <string> IN <field>
```

Usage

The `replace` operator can be used to normalize string values or enrich string values in results.

Examples

Example 1: Replace an IP address with a hostname in the destination field.

```
|where src="172.18.21.131" |replace "71.6.142.87" WITH "myhost"
IN dest
```

Example 2: Replace country name with abbreviation using exact match.

```
|iplocation src_ip | aggr count as cnt by src_ip, country, lat,
lon |replace "United Kingdom" WITH "UK" IN country
```

Example 3: Replace country name using regex pattern matching.

```
|iplocation src_ip | aggr count as cnt by src_ip, country, lat,
lon |regex_replace "United Kingdom" WITH "UK" IN country
```

Example 4: Counts Fortigate Firewall events, then replaces the full product name with the abbreviation `FW` in the `metadata.product.name` field.

```
ocsf | where metadata.product.name = 'Fortigate Firewall' | aggr
count by metadata.product.name | regex_replace 'Fortigate
Firewall' WITH 'FW' IN metadata.product.name
```

return

Use the `return` operator to return values from a subsearch.

Syntax

`|return [<count>] [<field>,...]`

`[<count>]`	Number of rows to return. If count is not specified, 1 row is returned.
`[<field>...]`	Comma-separated list of fields to return. With this option, field-value pairs are returned in the form <field>=<value>. For example, if you specify `|return user`, results have the form `user=doc-user@example.com`.

`[<alias>=<field>...]`	Comma-separated list of fields to return, replacing the field name with the specified alias. With this option, field-value pairs are returned in the form <alias>=<value>. For example, if you specify `|return username=user`, results have the form `username=doc-user@example.com`.
`[$<field>...]`	Comma-separated list of field values to return. With this option, only values are returned in the form <value>. For example, if you specify `|$user`, results have the form `doc-user@example.com`.

Usage

You can use the `return` operator to return values from a subsearch. It replaces the fields specified as arguments with one result field named `search`.

You can specify any combination of field list options: the field-value pair option, the alias-value pair option, or the values-only option.

Examples

Example 1: Return specified fields from subsearch as field-value pairs.

```
|return id, ip, user
```

Example 2: Return three rows with mixed field-value and value-only formats.

```
|return 3 id, $ip, user
```

Example 3: Selects key event identity fields, sorts by hostname, then returns only those three fields as the final output.

```
ocsf | fields metadata.uid,device.hostname, class_name | sort
device.hostname | return metadata.uid,device.hostname, class_name
```

reverse

Use the `reverse` operator to reverse the order of the results.

Syntax

`|reverse`

Usage

You can use the `operator` operator to reverse the order of results. The reverse operator works in similar way to the sort operator, but it does not require any parameters. It does not affect or modify the previous results, only the order of it.

Examples

Example 1: Reverse the order of results.

```
|reverse
```

Example 2: Aggregate action counts, sort, and then reverse the order.

```
|aggr count as nb_actions by action |sort nb_actions |reverse
```

Example 3: Sort by multiple fields with mixed sort directions and reverse.

```
|sort field1 asc, field2 desc | reverse
```

Example 4: Reverse results after sorting by three fields with varied directions.

```
|sort field1 desc, field2 desc, field3 asc | reverse
```

Example 5: Counts events by activity type, sorts ascending by count, then reverses the row order.

```
ocsf | aggr count as nb_actions by activity_name | sort nb_
actions | reverse
```

rex

Use the `rex` operator to create one or more new fields or rewrite one or more fields with values populated by parsing an existing field with a Perl compatible regular expression (PCRE) or sed pattern.

Syntax

> **Note:** You must place the rex operator **after** all condition clauses, to minimize the number of records processed. See AQL Best Practices for more information.

```
|rex [mode=sed] field=<field> "<regex or sed
<field>>"
```

`mode-sed`	Specify this option to use a sed expression instead of PCRE.
`field=<field>`	Specify the event log field to be parsed. For example, the following specifies that the url field is parsed: `field=url`
`"<<regex or sed <field>>"`	Specify the regex or sed expression. Use angle brackets to specify the part of the expression that is the extracted field. For example, the following specifies that a new field named hostname is extracted: `"https://(?<hostname>.*)/.*"`

Usage

You can use the `rex` operator to extract new fields using a regular expression. In the examples, the new field `hostname` is extracted from the `url` field; the new field `dst` is extracted from the `dest` field.

Example

Example 1: Extract agent message and raw event from message field for Cisco sourcetype.

```
|where sourcetype = 'cisco' AND action = 'create'

|rex field=message '.*?msg=(?<agent_msg>.*?)rawEvent=
(?<rawEvent>.*?)$'

|fields action, event_time, category, sourcetype
```

Example 2: Extract hostname from URL field using PCRE pattern.

```
|rex field=url "https://(?<hostname>.*)/.*"|where url is not null
```

Example 3: Extract destination hostname prefix from destination field.

```
|rex field=dest "^(?<dst>.*?)[\.|$]" |where dst is not null
|fields dest, dst
```

Example 4: Extracts the event time Unix timestamp from raw Fortinet log strings using a regex capture group, then converts it to a parsed datetime field.

```
ocsf

| where metadata.product.vendor_name contains 'Fortinet' and raw_
data contains 'eventtime='

| fields raw_data

| rex field=raw_data 'eventtime=(?&lt;unix_timestamp&gt;\\d+)'

| calc parsed_time=strptime(unix_timestamp, '%s')
```

rmvtag

Use the `rmvtag` operator to remove a tag or multiple tags to ThreatStream observables for the IOCs returned by an ioc or iocmatch lookup table search.

Syntax

```
|rmvtag <tagnames> [is_public=true|false]
```

`<tagnames>`	Comma-separated list of tag names to remove from the IOCs.
`[is_public]`	Specify `true` to remove public tags or `false` to remove private tags. If this option is not specified, the implicit default is `false`.

Usage

You can use the `rmtag` operator to remove a tag or multiple tags from ThreatStream observables for the IOCs returned by an ioc or iocmatch lookup table search. For example, you can perform a search on the ioc lookup table to find indicators with a confidence score of less than 10, low severity, and that has not been modified in the past 3 months; and then pipe this operator to remove a specific tag the organization uses to identify indicators that are of high concern and routinely sent to Integrator destinations.

In addition to removing the tag in ThreatStream, the operator adds a column to search results named `tag_removed` with the value `True`.

For information about ThreatStream observable tags, refer to ThreatStream online help.

Examples

Example 1

```
ioc|rmvtag "tag_name_1"
```

Example 2

```
ioc|rmvtag "tag_name_1, tag_name_1" is_public=false
```

sandbox

Use the `sandbox` operator to submit a URL to a ThreatStream sandbox.

Syntax

`|sandbox <field or value> '<sandbox_type>'`

`<field or value>`	Specify a field having a URL; or specify a URL.
`<sandbox_type>`	Specify the type of sandbox configured in ThreatStream for your organization: • `ANOMALISANDBOX` • `VMRAY` • `JOESANDBOX`

Usage

You can use the `sandbox` operator to submit a URL to a ThreatStream sandbox. In addition to submitting the URL, the operator adds the following fields to results:

- `sandbox_submission_success`: `1` indicates submission successful; `0` indicates submission failed.
- `sandbox_error_message`: String containing an error message (if any).

For information about sandboxes, refer to ThreatStream online help.

Examples

Example 1: Submit URL field to VMRAY sandbox for analysis.

```
|sandbox url 'VMRAY'
```

Example 2: Submit hardcoded URL to Anomali sandbox for analysis.

```
|sandbox 'https://www.fakeurl.com/img_1.png' 'ANOMAL
```

savedsearch

Use the `savedsearch` operator to run a previously saved search or use a previously saved search as a template that allows substitution of one specified string.

Syntax

```
|savedsearch "<search_name>" [<replace_
string>='<substitute_string>']
```

`<search_name>`	Specify the saved search to run. Surround the saved search name argument in single or double quotes. The argument is not case sensitive.
`[<replace_string>='<substitute_string>']`	If you want to replace a string argument in the saved search, specify a key-value pair. The key represents the string you want to replace and the value represents the substitute value. For example, if you specify `replace_me='hello'`, the replace string functions looks for the value `$replace_me$` and substitutes the string `hello`. Surround the substitution value in single or double quotes.

Usage

You can use the `savedsearch` operator to run a saved search from the AQL search box. You have two options:

- Run the saved search syntax precisely.
- Run the saved search syntax with one string substitution. In essence, you use the saved search as a template where you specify a substitution string value for the saved string value.

You can specify only saved searches belonging to your organization.

Examples

Example 1: Execute a saved search by name.

```
|savedsearch "save_test"
```

Example 2: Execute saved search with string substitution.

```
|savedsearch "save_replace_test" replace_me="I have been
replaced!"
```

score

Use the `score` operator to execute statistical tests to verify the results produced by machine learning models.

Syntax

```
|score <scoring-method> <a_field_1, a_field_2, ...
a_field_n> against <b_field_1, b_field_2, ... b_
field_m> [normalize={true|false}]
```

`<scoring-method>`	Specify a scoring method: • `accuracy_score` Use accuracy scoring to get the prediction accuracy between actual-labels and predicted-labels. This method is the sklearn.metrics.accuracy_score from scikit-learn. Refer to scikit-learn documentation for complete details: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html • `confusion_matrix` Use the confusion matrix to evaluate the accuracy between actual-labels and predicted-labels. This method is the sklearn.metrics.confusion_matrix method from scikit-learn. Refer to scikit-learn documentation for complete details: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html • `mean_absolute_error` Use the mean absolute error scoring to calculate regression loss between actual_fields and predicted_fields. This method is the sklearn.metrics.mean_absolute_error. method from scikit-learn. Refer to scikit-learn documentation for complete details: https://scikit-https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html#sklearn.metrics.mean_absolute_error • `mean_squared_error` Use the mean squared error score to calculate regression loss between actual_fields and predicted_fields. This method is the sklearn.metrics.mean_squared_error method from scikit-learn. Refer to scikit-learn documentation for complete details:

	https://scikit-https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html#sklearn.metrics.mean_squared_error
`<x_field_n> against <x_field_n>`	Comma-separated list of fields to compare.
`[normalize={true|false}]`	The `accuracy_score` method supports an option to specify your data normalization preference: • `true`: Return the fraction of correctly classified samples. • `false`: Return the raw count of correctly classified samples. The implicit default if this option is not specified is `true`.

Usage

You can use the `score` operator to execute statistical tests to verify the results produced by machine learning models. The `score` operator is a versatile tool that can be applied across various use cases to validate models and their statistical outcomes. Depending on the nature of your data and the specific problem you aim to address, you can select the most appropriate scoring method provided by the score command.

The accuracy_score method supports pairwise comparisons between two sets of fields.

The other supported methods are used to compare the actual value and the predicted value.

Examples

Example 1: Calculate accuracy score with raw count of correctly classified samples.

```
|score accuracy_score a1,a2 against p1,p2 normalize=false
```

Example 2: Calculate accuracy score as fraction of correctly classified samples.

```
|score accuracy_score a1,a2 against p1,p2 normalize=true
```

Example 3: Generate confusion matrix for actual and predicted values.

```
|score confusion_matrix actual_value against predicted_value
```

Example 4: Calculate mean absolute error between actual and predicted values.

```
|score mean_absolute_error actual_value against predicted_value
```

Example 5: Calculate mean squared error between actual and predicted values.

```
test_score_table | score mean_squared_error actual against
predicted
```

Example 6: Filters and cleans network flow fields, fills nulls with defaults, sorts by most recent event, then calculates the mean absolute error between source and destination port numbers.

```
ocsf | where severity in ('alert', 'info') | fields severity,
src_endpoint.interface_name, src_endpoint.port, dst_
endpoint.interface_name, dst_endpoint.port, src_endpoint.ip, dst_
endpoint.ip | fillnull src_endpoint.interface_name value='no
source' | fillnull src_endpoint.ip, dst_endpoint.ip | fillnull
dst_endpoint.interface_name value='empty_dest' | sort event_time
desc | score mean_absolute_error src_endpoint.port against dst_
endpoint.port
```

scrub

Use the `scrub` operator to replace sensitive data with generated fictional values.

Syntax

`|scrub [user_data_type](field>) [...]`

`[user_data_type]`	Optional. Specify the type of data to generate as replacement data: `address`, `city`, `company`, `country`, `credit_card_expire`, `credit_card_number`, `date_of_birth`, `date_time`, `domain_name`, `email`, `file_extension`, `file_name`, `first_name`, `ipv4`, `ipv6`, `job`, `last_name`, `latitude`, `longitude`, `mime_type`, `name`, `number`, `paragraph`, `phone_number`, `random_digit`, `random_letter`, `sentence`, `text`, `url`, `user_name`, `uuid4`, `word` If you do not specify a data type, the implicit default is `word`. In other words, if you do not specify a type, the replacement data is a random word.
`<field>`	Required. Specify the field to scrub.

Usage

You can use the `scrub` operator to anonymize sensitive data in search results. Sensitive data might include user names, IP addreses, domain names, and so on. For example, when used to anonymize user names, the `scrub` operator might transform the value `user=maria@anomali.com` into the fictional value `user=anna@example.com`. The transformation enables Anomali Security Analytics users to share search results without revealing confidential or personal information.

You can use the `scrub` operator in conjunction with other operators—for example, after you have filtered the search and refined result columns.

You can specify multiple fields to scrub. Refer to the examples.

Examples

Example 1: Replace user field with fictional email and URL field with fictional URL.

```
|scrub email(user) url(url)
```

Example 2: Replace user, URL, source IP, and sourcetype fields with fictional data.

```
|scrub email(user) url(url) ipv4(src) sourcetype
```

Example 3: Takes 10 OCSF events with user, URL, hostname, and product fields, then anonymizes `actor.user.name` as an email format and redacts `metadata.product.name`

```
ocsf | fields actor.user.name, http_request.url.url_string, src_
endpoint.hostname,metadata.product.name | limit 10 | scrub email
(actor.user.name) metadata.product.name
```

selfjoin

Use the `selfjoin` operator to join a table with itself by combining each row with the next row in the results set based on the specified join field.

Syntax

`|selfjoin <join_field>`

`<join_field>`	Table field to perform the selfjoin on.

Usage

You can use the `selfjoin` operator to combine and correlate related information—for example, when a result set contains some parent task information and child task information.

Examples

Example 1: Join table rows based on matching host field values.

```
|selfjoin host
```

Example 2: Join table rows based on matching source IP field values.

```
|selfjoin src_ip
```

Example 3: Joins the OCSF dataset against itself on metadata.product.name, pairing rows that share the same product.

```
ocsf | selfjoin metadata.product.name
```

set

Use the `set` operator to generate a table from two subsearches with a set operation applied to search results.

Syntax

`|set <operation> <subsearch1> <subsearch2>`

`<operation>`	Required. Specify the operation to be applied to search results. Possible options: • `intersect`: The resulting table will contain the results that exist in both subsearches. • `union`: The resulting table will contain the results from both subsearches. The ones that exist in both subsearches will be presented only once. • `diff`: The resulting table will contain the results from both subsearches. The ones that exist in both subsearches will be excluded.
`<subsearch1>`	Required. Specify the first subsearch, using square brackets (`[]`).
`<subsearch2>`	Required. Specify the second subsearch, using square brackets (`[]`).

Usage

You can use the `set` operator to find differences or similarities between the result sets in a consolidated form.

Examples

Example 1: Find differences between production and staging event logs.

```
|set diff [eventlog | where sourcetype = 'prod' | fields src,
dest, timestamp, bytes_in] [eventlog | where sourcetype =
'staging' | fields src, dest, timestamp, bytes_in]
```

Example 2: Find common entries between CrowdStrike and Netscope event logs.

```
|set intersect [eventlog | where sourcetype contains
"crowdstrike" | fields src, dest] [eventlog | where sourcetype
contains "netscope"| fields src, dest]
```

setfields

Use the `setfields` operator to create new fields and string values.

Syntax

```
|setfields <name>="value"[, <name>="value"], ...
```

`<name>="value"`	Specify field name-value pairs. You can specify a comma-separated list of such name-value pairs. Only string values are supported.

Usage

You can use the `setfields` operator to pass in one or more name-value pairs to create new fields. The value must be surrounded by quotation marks. The new field or fields are added to each row returned by the search.

Examples

Example 1: Create multiple new fields with string values.

```
|setfields test_field1="hello1", test_field2='hello2'
```

Example 2: Selects `activity_name` from OCSF events then adds two constant string fields `test_field1` and `test_field2` with static values.

```
ocsf | fields activity_name | setfields test_field1='hello1',
test_field2='hello2'
```

shodan_database

Use the `shodan_database` operator to enrich results related to an IP address or field containing an IP address using the ThreatStream Shodan enrichment.

Syntax

`|shodan_database <field or value>`

`<field or value>`	Specify the field name or IP address to look up.

Usage

You can use the `shodan_database` operator to enrich results with data from the ThreatStream Shodan integration.

Examples

Example 1: Enrich IP field with Shodan database information.

```
|fields ip |shodan_database(ip)
```

Example 2: Enrich a hardcoded IP address with Shodan data.

```
|shodan_database '192.0.2.1'
```

sort

Use the `sort` operator to sort the results table by the specified field, in ascending or descending order.

> **Caution:** To correctly sort different columns, use the sort operator in your AQL query instead of using the sort icon next to the column name.

Syntax

```
|sort <field> [asc|desc]
|sort by <field> [asc|desc]
```

Usage

You can sort by one or more fields. Ascending order (low to high) or descending order (high to low). The alternative syntax shown above are equivalent.

Examples

Example 1: Sort field values in ascending order.

```
|sort sourcetype asc
```

Example 2: Sort field values in descending order using alternative syntax.

```
|sort by num_column desc
```

Example 3: Selects key OCSF fields, drops `src_endpoint.hostname`, then sorts all results alphabetically by product name.

```
ocsf | fields event_time, metadata.product.name, src_
endpoint.hostname, activity_name | fields - src_endpoint.hostname
| sort metadata.product.name
```

snow

Use the `snow` operator to create or update an investigation in ServiceNow.

Syntax

`|snow <field or ID>`

`<field or ID>`	ThreatStream investigation ID field or ThreatStream investigation ID.

Usage

You can use the `snow` operator to create or update an investigation in ServiceNow, provided ServiceNow has been integrated with your ThreatStream organization. The operator adds `incident ID`, `incident sys_ID`, and `incident URL` to results.

Examples

Example 1: Create or update ServiceNow incident with investigation ID number.

```
|snow 123456
```

Example 2: Create or update ServiceNow incident with quoted investigation ID.

```
|snow '123456'
```

Example 3: Create or update ServiceNow incident with double-quoted investigation ID.

```
|snow "123456"
```

Example 4: Create or update ServiceNow incident using investigation ID field.

```
|calc ts_investigaion_id = 123456 | snow ts_investigaion_id
```

snowevent

Use the `snowevent` operator to create an event in ServiceNow.

Syntax

```
|snowevent snowaccount="<string>"
snownode="<string>" snowresource="<string>"
snowtype="<string>" snowseverity=<number>
[snowdescription="<string>"]
```

`snowaccount="<string>"`	Specify the ServiceNow account user name.
`snownode="<string>"`	Specify a string to populate the ServiceNow event source.
`snowresource="<string>"`	Specify a string to populate the ServiceNow resource name.
`snowtype="<string>"`	Specify a string to populate the ServiceNow resource type.
`snowseverity=<number>`	Specify a number to populate the ServiceNow severity.
`[snowdescription="<string>"]`	Optional. Specify a string to populate the ServiceNow description.

Note: String values must be enclosed in double-straight quotes (”). If the value itself must have a double-quote, precede the

> double-quote you want to preserve with a backslash (\).

Usage

You can use the `snowevent` operator to create an event in ServiceNow. The query results table includes a column named `service_now_event_link`, which contains the URL that you can use to navigate to the event in ServiceNow.

> **Note:** Anomali recommends you do not use this operator with very large tables. Creating a large number of ServiceNow events can be slow. Use the `limit` operator to limit results, as shown in the example.

Examples

Example 1: Create ServiceNow event with account, node, resource, type, and severity information.

```
[...]|limit 1 |snowevent snowaccount="ACCOUNT"
snownode="localhost" snowresource="CPU" snowtype="Virtual
Machine" snowseverity=2 snowdescription="CPU
```

snoweventstream

Use the `snoweventstream` operator to create an event in ServiceNow.

Syntax

```
|fields node, resource, eventtype, severity,
description |snoweventstream
```

`node="<string>"`	Specify a string to populate the ServiceNow event source.

`resource="<string>"`	Specify a string to populate the ServiceNow resource name.
`eventtype="<string>"`	Specify a string to populate the ServiceNow resource type.
`severity=<number>`	Specify a number to populate the ServiceNow severity.
`[description="<string>"]`	Optional. Specify a string to populate the ServiceNow description.

> **Note:** String values must be enclosed in double-straight quotes (\"). If the value itself must have a double-quote, precede the double-quote you want to preserve with a backslash (\\).

Usage

You can use the `snoweventstream` operator to create an event in ServiceNow. The query results table includes a column named `service_now_event_link`, which contains the URL that you can use to navigate to the event in ServiceNow.

> **Note:** Anomali recommends you do not use this operator with very large tables. Creating a large number of ServiceNow events can be slow.

Examples

Example 1

```
deletemodel_test
| calc node="host"
| calc resource="ram"
| calc eventtype="warning"
| calc severity=5
```

```
| calc description="ram memory is full"
| snoweventstream
```

snowincident

Use the `snowincident` operator to create an incident in ServiceNow.

Syntax

```
|snowincident [snowcategory=<string|field>]
[snowshort_description=<string|field>]
[snowcontact_type=<string|field>]
[snowurgency=<integer|field>]
[snowsubcategory=<string|field>]
[snowstate=<string|field>]
[snowlocation=<string|field>]
[snowimpact=<integer|field>]
[snowpriority=<integer|field>]
[snowassignment_group=<string|field>]
[snowcomments=<string|field>]
[snowexternal_url=<string|field>]
[snowcorrelation_id=<string|field>]
[snowcustom_fields=<string|field>]
```

`snowcategory=<string|field>`	Optional. Specify a ServiceNow incident category, for example, `Phishing`, `Policy violation`.
`snowshort_description=<string|field>`	Optional. Specify a short incident description.
`snowcontact_type=<string|field>`	Optional. Specify a preferred contact type, for example, `Phone`, `Email`.

`snowurgency=<integer|field>`	Optional. Specify an incident urgency.
`snowsubcategory=<string|field>`	Optional. Specify an incident subcategory, for example, `Spear phishing`.
`snowstate=<string|field>`	Optional. Specify an incident state, for example, `Analysis`.
`snowlocation=<string|field>`	Optional. Specify an incident location, foe example, `California`.
`snowimpact=<integer|field>`	Optional. Describe the incident's impact on the business.
`snowpriority=<integer|field>`	Optional. Specify an incident priority.
`snowassignment_group=<string|field>`	Optional. Specify a preferred assignment group.
`snowcomments=<string|field>`	Optional. Specify additional comments separated with a comma.
`snowexternal_url=<string|field>`	Optional. Specify URLs related to the incident.
`snowcorrelation_id=<string|field>`	Optional. Specify an incident correlation ID.

`snowcustom_fields=<string|field>`	Optional. Specify custom fields and their values separated with the `||` symbol. All custom fields must be configured in the ServiceNow instance first. The parameters passed in custom fields must be configured in the ServiceNow Incident table and the appropriate transform map in ServiceNow. For example, `u_affected_user` and `u_caller_id` must be present in your ServiceNow Incident table.

Usage

You can use the `snowincident` operator to create ServiceNow incidents. As a result, you will get the Incident Link column containing the URL that you can use to navigate to the incident in ServiceNow.

Default values of the arguments depend on the default values that are configured in your ServiceNow instance. For example, if the **Comment** field in ServiceNow is configured to be an empty string by default, then the default value for the `snowcomments` argument will be an empty string as well.

> **Note:** Using the operator with the same set of constant arguments one more time during single transaction will not create another ServiceNow incident.

> **Note:** Avoid using this operator with large tables as creating a large number of events may take long.

Examples

Example 1: Create a ServiceNow incident with comprehensive field values.

```
|limit 1 | snowincident snowcategory="Denial of Service"
snowshort_description="test description" snowcontact_type="Phone"
snowurgency=1 snowsubcategory="Inbound DDos" snowstate="Analysis"
snowlocation="New Castle" snowimpact=1 snowpriority=5
snowassignment_group="Security Incident Assignment"
snowcomments="test comment
```

Example 2: Takes 2 OCSF events and creates a ServiceNow incident for each, populating category, urgency, impact, priority, assignment group, location, and custom fields.

```
ocsf | limit 2 | snowincident snowcategory='Denial of Service'
snowshort_description=activity_name snowcontact_type='Phone'
snowurgency=1 snowsubcategory='Inbound DDos' snowstate='Analysis'
snowlocation='New Castle' snowimpact=1 snowpriority=5
snowassignment_group='Security Incident Assignment'
snowcomments='test comment' snowexternal_
url=metadata.product.name snowcorrelation_id='1' snowcustom_
fields='caller=Meredith Ivrin'
```

spath

Use the `spath` operator to read in information from an XML or JSON file specified by a location path expression.

Syntax

```
|spath [input=<field>] [output=<field>]
[path=<datapath>|<datapath>]
```

`[input=<field>]`	Specify the field from the source file to read in.
`[output=<field>]`	Specify the field to write to in search results. If you do not specify the output field name, the field name is derived from the specified location path.

`[path=<datapath>|<datapath>]`	Specify the location path. You can explicitly specify the `path=` option or not. If it is not specified, AQL can still interpret a path specified after the field arguments as a path argument.
	Use a dot-separated hierarchy to specify the location steps in the datapath.
	A location step is a field name and optional index that refers to the position of the data in an array. For example: `parent.child{2}` refers to the second child in the hierarchy.
	Use the `@` symbol to specify an XML attribute. For example: `parent.child{@type}` identifies the XML location for information that distinguishes children of adult age and minor children:
	`<parent>`
	`  <child age="adult">Alice</child>`
	`  <child age="minor">Bob</child>`
	`</parent>`

Usage

You can use the `spath` operator to read in information from an XML or JSON file.

Examples

Example 1: Extract a nested field from JSON or XML data using explicit path syntax.

```
|spath output=locDesc path=vendorProductSet.product.desc.locDesc
```

Example 2: Extract a nested field from a specified input field.

```
|spath input=json_field output=myfield
vendorProductSet.product.desc
```

Example 3: Extract nested data with implicit path syntax.

```
|spath input=json_field output=myfield event.sou
```

Example 4: Counts daily Fortinet product events, then pivots the result so each date becomes a column and each row represents a product.

```
ocsf | where metadata.product.name contains 'Forti' | calc
Date=strftime(event_time, '%Y-%m-%d') | aggr count as cnt by
metadata.product.name, Date | transpose row=metadata.product.name
col=Date
```

streamstats

Use the `streamstats` operator to generate cumulative statistics for each event in your data, streaming up to the current event. In other words, it adds new fields to your current results as it performs the specified statistics at row level, while accumulating the results from all previous events up to the current event in a specified time window.

This operator works with both `eventlog` and `ocsf` schemas:

- For the `eventlog` table, events are ordered based on `event_time` and `event_meta_id` parameters.
- For the `ocsf` table, events are ordered based on `event_time` and `event_seq_id` parameters to ensure proper sequential ordering across OCSF event classes.
- For other tables, if no ordering is available, an internal row number is used to maintain ordering and statistics are calculated based on this.

When used with OCSF fields, `streamstats` supports nested object fields using dotted notation. Examples include `metadata.product.name` and `actor.user.name`.

> **Caution:** `streamstats` performs expensive aggregation operations. Therefore, we recommend using this command on shorter time periods. Searches running over very large time ranges using `streamstats` can result in searches timing out.

Syntax

```
| streamstats <aggregation-list> [by <field-list>]
[current=<boolean>] [window=<integer>]
[limit=<integer>]
```

`<aggregation-list>`	Represents a comma-separated list of aggregations to be included in the result. For example: `count,sum(event_size)`. List of supported aggregations: • `count` • `sum` • `avg` • `min` • `max` • `range` • `previous`: returns the previous value of the event in the specified window • `next`: returns the next value of the event in the specified window • `dc`: returns a count of distinct values within the specified window

`by <field-list>`	Specifies the list of fields to include in the group-by clause, Similar to the `aggr` operator, the `streamstats` operator allows you to create aggregations (statistics) at the group level. In this case, the aggregations are calculated for each group in the list of fields.
`current=<boolean>`	Specifies if the current row must be considered for the calculations. Setting it to `False` selects only the previous rows while excluding the current row. Default: `True`.
`window=<integer>`	Specifies the number of events to be used when calculating the statistics. This window size lets you: • Consider only some rows for the calculation, and • Configure how many previous events you wish to include for the calculation. Default: `0`, which means, all previous events are included in the calculation. **Note:** When using `dc`, `window` is required and must be between `1` and `50`. An unbounded window (`window=0`) is not supported with `dc`.
`limit=<integer>`	Specifies the limit of the number of events to be processed. Default: `1000000`.

Usage

You can use the `streamstats` operator to make running calculations on your events. For example, you can count the number of events by any field or to get the previous value of the field to make any further calculations.

Examples

Example 1: Eventlog with running statistics

```
| streamstats count as cnt, sum(bytes_in), next(bytes_in),
previous(bytes_in) by app_type
```

Example 2: Multiple streamstats operations

```
| streamstats count as cnt, avg(bytes_in) by app_type |
streamstats count as cnt_total
```

Example 3: Windowed streamstats

```
| streamstats count as cnt, next(bytes_in) by sourcetype window=5
current=false
```

Example 4: OCSF activity tracking with previous value

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

```
ocsf | streamstats previous(activity_name) as pr_activity |
fields event_time, activity_name, metadata.product.name, pr_
activity | sort event_time
```

Example 5: Distinct count over a sliding window

```
| streamstats dc(src) by sourcetype window=10
```

superapi

Use the `superapi` operator to perform GET, POST, PUT, PATCH, and DELETE requests to internal ThreatStream API endpoints directly from the AQL Search interface and without additional authentication. The operator enables automation of

data retrieval, updates, and bulk operations on ThreatStream. It supports flexible input specifications using fields from your pipeline, and allows you to explicitly define output fields and their types for correct parsing.

Before you begin

Before using the `superapi` operator, consider the following:

Fetching live data

To fetch data on the fly, always use the `anomali_api_proxy` placeholder table with the `superapi` operator.

```
anomali_api_proxy | superapi
```

Resource consumption

Running queries with a large number of response rows or extended time ranges can lead to excessive resource consumption on the API server. To avoid service disruption, use the `superapi` operator cautiously and limit both the scope and frequency of your requests.

Response size limit

The `superapi` operator tracks the cumulative size of all HTTP responses within a single query.

When the total exceeds 50 MB, the operator stops making additional API calls. Rows processed before the limit is reached return normally. Rows that would have been processed after the limit is reached receive a warning in their output instead of API results.

Consider the following example:

```
example_actors_list

| fields name as actor_name, id

| superapi
endpoint='/api/v1/actor/<<id>>/tipreport/?limit=1
000' input_fields='id'
```

```
| calc objects=json_extract(response_data,
'objects', ARRAY)

| mvexpand objects

| calc name=json_extract(objects_expanded,
'name')

| calc modified_ts=json_extract(objects_expanded,
'modified_ts')

| calc created_ts=json_extract(objects_expanded,
'created_ts')

| fields name, actor_name, id, modified_ts,
created_ts

| where actor_name contains_ci
("APT27","APT28","APT34","APT41")

| sort modified_ts desc
```

In this query, the first line starts with a list of actors. Each row has an `actor_name` and an `id`. The `superapi` line then makes one HTTP API call per actor row, replacing `<<id>>` with the actor ID in that row.

For example, if the input table has the following rows:

Row	actor_name	id	API call
1	APT27	101	`/api/v1/actor/101/tipreport/?limit=1000`
2	APT28	102	`/api/v1/actor/102/tipreport/?limit=1000`
3	APT34	103	`/api/v1/actor/103/tipreport/?limit=1000`

Because the endpoint uses `limit=1000`, each call can return up to 1,000 threat bulletins for that actor. If those reports are large,

each response can be several MB, and could therefore exceed the 50 MB limit.

> **Note:** Because API calls are executed in parallel, the order in which rows are processed is non-deterministic. Once the limit is reached, the specific rows that are skipped may vary between runs.

When the limit is reached, Anomali Search displays the following warning:

```
API response size limit reached. Some results may
be incomplete due to large API responses.
```

Skipped rows contain the following in their output:

```
{"warning": "API response size limit reached. Row
skipped."}
```

Additionally, caching responses ensures completed requests are not reprocessed again for non-idempotent requests (POST/PATCH), preventing duplicate records.

Recommended Best Practice

If you see the limit warning, reduce the scope of your query using one or more of the following approaches:

- **Limit input rows.** Add `| limit N` before `| superapi` to cap the number of API calls made. For example:

  ```
  | makeresults count=100

  | streamstats count AS counter

  | limit 20
  ```

```
| superapi
endpoint='/api/v1/intelligence/?limit=1000&offs
et=<<counter>>' input_fields='counter'
```

- **Use a smaller page size.** Lower the `limit=` parameter in your endpoint URL to reduce the size of each individual response. For example, use `limit=100` instead of `limit=1000`.
- **Filter before the API call.** Add `| where` conditions before `| superapi` to reduce the number of rows that need API enrichment.

Syntax

```
superapi
    endpoint = <string>
    method = <string>
    payload | body_content= <string>
    output = <string>
    output_types = <string>
```

`endpoint`	Specifies the endpoint used to search on ThreatStream. For a list of available ThreatStream API endpoints, refer to the *ThreatStream API Reference Guide*, which can be downloaded on the Downloads page.
`method`	Specifies the method to perform the request to ThreatStream. Default: `get`

`payload` | `body_content`	The `payload` or `body_content` keywords are used to specify the payload to be sent in the request. When using the `payload` keyword, the value must be a valid JSON-formatted string. This payload will be included in the body of the request. The `body_content` keyword allows you to include input data in the request body using fields from the pipeline. You can reference these fields using the syntax `<<field>>`. For example, `body_content=' { "model_type" : "<<model_type>>", "name": "<<search_name>>"}'`

`output`	Specifies a comma-separated list of fields to extract from the response. These fields are included in the output of the operator.
	Key features:
	• **Full path support:** You can extract fields using their full path in the response structure.
	For example, if the response contains the following structure:
	`{ "objects": [ { "id": 1, "name": "test" } ] }`
	Then the following fields are equivalent:
	▪ `output='id'`
	▪ `output='objects.{}.id'`
	• **Alias Support:** You can rename output fields using the `as` keyword.
	For example,
	`output='id as identifier,name'`
	• **Extracting Fields from Nested Paths:** When the response contains nested objects, you can extract information from each key in the hierarchy.
	For example, if the response contains the following structure:

```
{
    "results": {
        "threat_model": {
            "created": {
                "actor": {
                    "2025-03-01": 1,
                    "2025-03-03": 41,
                    "2025-03-04": 126
                },
                "infrastructure": {
                    "2025-03-01": 57,
                    "2025-03-02": 49
                }
            }
        }
    }
}
```

You can extract every key in the hierarchy and get the count at the end of it in different output fields.

The syntax for this is the following:

- Path of the desired hierarchy
- `as` keyword to delimit the hierarchy and aliases
- Aliases for each field in the hierarchy.

The final output to get the data in the hierarchy is as follows:

	`output=threat_model.{}.{}.{} as status;type;event_time;cnt` This output generates 4 fields: • `status` • `type` • `event_time` • `cnt`
`output_types`	Specifies a comma-separated list of field types corresponding to each output field defined in the output parameter. If the number of `output_types` specified doesn't match the number of output fields or the type specified is not supported, then an error is displayed. **Supported TS API output types:** • string • integer • number • timestamp • boolean • list

Examples

Example 1: Retrieve the first 20 threat intelligence entries with a confidence greater than 80 and a severity of "low".

```
anomali_api_proxy | superapi

    endpoint = "/api/v2/intelligence?limit=20&&confidence__
gt=80&severity=low"
```

```
    method = get

    output = "objects.{}.id as id, objects.{}.uuid as uuid,
objects.{}.value as value, objects.{}.confidence as confidence,
objects.{}.severity as severity, objects.{}.created_ts as
created_ts"

    output_types = "number,string,string,number,string,timestamp"
```

Example 2: Retrieve observables with private tags.

```
anomali_api_proxy | superapi

    endpoint = "/api/v1/intelligence/?limit=1000&is_
public=false&tags.tlp__isnull=false"

    method = get

    output = "objects.{}.id as id, objects.{}.value as value,
objects.{}.itype as itype, objects.{}.confidence as confidence,
objects.{}.severity as severity, objects.{}.tags.{}.name as tag_
name, objects.{}.tags.{}.tlp as tag_tlp, objects.{}.created_ts as
created_ts"

    output_types =
"number,string,string,number,string,string,string,timestamp"
```

Example 3: Combine lookup table results from multiple API endpoints.

```
ocsf | superapi endpoint = '/api/v1/xdr/event/lookup/test_table_
iris' output='result.modified_ts as modified_ts, result.modified_
by_user as modified_by' | calc source='test_table_iris' |
appendtable [| superapi endpoint =
'/api/v1/xdr/event/lookup/test_table_iris_passthrough'
output='result.modified_ts as modified_ts, result.modified_by_
user as modified_by' | calc source='test_table_iris_passthrough']
```

Example 4: Calls the XDR IOC exclusion list API endpoint and maps the response fields into the result set for each OCSF event.

```
ocsf | superapi endpoint='/api/v1/xdr/excludelist/ioc' output='
{}.created_by as created_by,{}.created_ts as created_ts,
{}.modified_by as modified_by,{}.modified_ts as modified_ts,
{}.username as username,{}.value as value'
```

tail

Use the `tail` operator to show the last *n* results in the result set according to the sort order.

Syntax

```
|tail <n>
```

Usage

Similar to the Unix `tail` command.

For non-aggregation queries, if no sort order is specified, the `tail` operator culls the last *N* results falling in the time range by `event_time`.

For aggregations, if no sort order is specified, the `tail` operator draws the line at random.

Example

Example 1: Return the last ten results from aggregated results sorted in descending order.

```
|aggr count as cnt by sourcetype | sort cnt desc | tail 10
```

Example 2: Takes the last 10 OCSF events, packages `class_name` and `metadata.product.name` into a JSON object, then parses it back out into individual fields

```
ocsf | tail 10 | fields class_name, metadata.product.name |
makejson class_name, metadata.product.name output=new_field |
fromjson new_field
```

timechart

Use the `timechart` operator to perform aggregations over timespans.

Syntax

```
|timechart <aggr_func> [as new_field] by [<field_to_
split_aggregate>] [span=<interval_with_time_unit>]
[limit=<number_of_columns>] [makecontinuous=<create_
empty_buckets>] [addtotals=<aggregate_totals>]
```

`<aggr_func>`	Required. Specify a function to calculate statistics. For example, `avg`, `count`, `count(distinct <field>)`, `max`, `min`, `sum`.
`<field_to_split_aggregate>`	Optional. Specify the field you want to aggregate. Each distinct value of the split-by field becomes a series in the chart. For example, you can split and aggregate on `sourcetype` to understand event trends for a particular data source over a specific interval.

`<interval_with_time_unit>`	Optional. Specify the time interval for the split. Multiple units are not allowed for week, month, or year. In other words, specify `1` as the interval integer when using week, month, or year. Possible values: `s`: time interval in seconds `m`: time interval in minutes `h`: time interval in hours `d`: time interval in days `w`: time interval in weeks `mon`: time interval in months `y`: time interval in years For example, `span = 1h` splits and aggregates data in one hour time buckets.
`<number_of_columns>`	Optional. Specify how many columns will be displayed in the results table. When using `<by field>` in the AQL query, if there are more values than the specified limit, the results that exceed the limit will be grouped in an additional column called `others`. Default: `10`.

`<create_empty_buckets>`	Optional. Specify whether to create empty buckets for the time range and span that contains no data. Possible values: • `true`: empty buckets are created for periods in the selected time range that have no data. Applicable to time units: `s` (second), `m` (minute), `h` (hour), and `d` (day). • `false`: no empty buckets are created for time range periods with no data. Default: `false`.

`<aggregate_totals>`	Optional. When using `<by field>` in the AQL query, specify whether to display the total for a field that is aggregated across the specified time range, bucketed by the split interval. Useful for comparing entities and spotting volume trends, without needing a separate aggregation query. When set to true, it automatically appends a new `total` row and column to the search results such that: • the `total` row aggregates the specified field across the full time range. • the `total` column is a sum across all values of the specified field. For example, setting `addtotals=true` on `sourcetype` for `span=1h` adds a row that is an aggregate across the full time range per sourcetype and a column that is a sum across all sourcetypes for each row.

Usage

The `timechart` operator can be used to aggregate data for a field to produce a chart, with time used as the x-axis. You can specify a split-by field, where each distinct value of the split-by field becomes a series in the chart.

> **Note:** After you issue a search with `timechart`, the value for `span` may be recalculated and rationalized to create meaningful buckets given the configured time range. Specify

> `makecontinuous=true` to create empty buckets for time periods with no data, for time units in `s` (seconds), `m` (minutes), `h` (hours), or `d` (days).

Examples

Example 1: Counts events over time, grouped by sourcetype.

```
| timechart count as cnt by sourcetype
```

Example 2: Counts events per hour, grouped by sourcetype.

```
| timechart count by sourcetype span=1h
```

Example 3: Counts events over time, showing only the top five sourcetypes, with others grouped.

```
| timechart count by sourcetype limit=5
```

Example 4: Filter AWS CloudTrail logs and chart hourly event counts with continuous time buckets.

```
| where sourcetype contains_ci 'aws_cloudtrail' | timechart count
by sourcetype span=1h makecontinuous=true
```

Example 5: Count hosts per source type in hourly buckets, creating the `total` row and column to summarize across all source types and time periods.

```
| timechart count(host) as devicecount by sourcetype span=1h
addtotals=true
```

Example 6: Create a time chart showing OCSF network activity trends.

```
ocsf | where class_name = "Network Activity" | timechart count by
connection_info.protocol_name span=1h
```

Example 7: Generates an hourly time series chart of OCSF event counts for the top 5 products.

```
ocsf | timechart count by metadata.product.name span=1h limit=5
```

timetrend

Use the `timetrend` operator to perform aggregations over timespans.

Syntax

```
|timetrend [columnname=<field>] <aggr_func> [as new_
field] [, <aggr_func> [as new_field], ...] [by
<field>, ...] [span=interval]
```

`columnname=<field>`	Optional. Specify a new time column other than event_time (default) for aggregation.
`<aggr_func>`	Statistics functions: `avg`, `count`, `count(distinct <field>)`, `max`, `min`, `sum`
`span= [<integer>][s,h,m,d,w,mon,y]`	Multiple units are not allowed for week, month, or year. In other words, specify `1` for the integer place when specifying week, month, or year.

Usage

The `timetrend` operator can be used with time series visualizations to show trends.

> **Note:** After you issue a search with `timetrend`, the value for `span` may be recalculated and rationalized to create meaningful buckets given the configured time range.

Examples

Example 1: Count events aggregated by month.

```
|timetrend count span=1mon
```

Example 2: Count events by source type with a ten-minute time span.

```
|timetrend count by sourcetype span=10m
```

Example 3: Count events using a custom timestamp field for aggregation.

```
|timetrend columnname=timestamp count by sourcetype
```

Example 4: Buckets OCSF events into two hour intervals, counts events per bucket, and sorts chronologically.

```
ocsf |timetrend count span = 2h | sort event_time asc
```

tojson

Use the `tojson` operator to convert event field data into JSON objects.

Syntax

```
|tojson [<tojson-function>] [default_
type=<datatype>] [fill_null={true|false}] [include_
internal={true|false}] [output_field=<string>]
```

[<tojson-function>]	Specify a data type conversion function to apply to the specified field. The function has the following syntax: [auto | bool | json | none | num | str] (<field>) The auto function automatically determines the output data type; the none function does not convert the data type; the other functions convert the data to their respective data types. If the function is not specified, the implicit default is none. If you do not specify field names, the implicit default is * (all fields). You can specify one or more tojson functions. Separate multiple functions with a space. Refer to the examples.
[default_type=<datatype>]	Specify the data type to apply to fields when the fields are not specifically associated with the data type function. If this option is not specified, no default data type is applied.
[fill_null={true|false}]	Specify true to populate the output field with the value null when the tojson search processor parses a field that does not have proper JSON formatting. If this option is not specified, the implicit default is false, meaning the tojson search processor skips the field.
[include_internal={true|false}]	Specify true to process internal fields, such as event_size. If this option is not specified, the implicit default is false, meaning internal fields are not included.
[output_field=<string>]	Specify the name of the field to which the tojson search processor writes the output JSON objects. If this option is not specified, the tojson search processor creates a field named _raw.

Usage

You can use the `tojson` operator to create a new field having a JSON value. You can use the `tojson` operator in conjunction with other operators—for example, after you have filtered the search and refined result columns.

Examples

Example 1: Convert all fields to JSON format.

```
|tojson
```

Example 2: Convert specified fields with explicit data type functions.

```
|tojson str(bytes_in) num(dcid)
```

Example 3: Filter fields and convert to JSON with type conversion.

```
|where bytes_in is not null |fields bytes_in, dcid|tojson str
(bytes_in) num(dcid)
```

Example 4: Apply a default data type to unspecified fields.

```
|tojson src sourcetype dest_ip default_type=num
```

Example 5: Populate null values for fields with improper JSON formatting.

```
|tojson src sourcetype dest_ip fill_null=true
```

Example 6: Include internal fields in JSON output.

```
|tojson src sou
```

Example 7: Serializes `src_endpoint.hostname`, `metadata.product.name`, and `metadata.uid` into a single JSON object per row, filling any null values, and stores the result in `result_json`.

```
ocsf | tojson src_endpoint.hostname metadata.product.name
metadata.uid fill_null=True output_field=result_json
```

top

Use the `top` operator to return the values with the top counts for the specified field.

Syntax

```
|top [n] <field> [,<field>, ...]
```

Usage

The `top` operator is equivalent of aggregating by a field, sorting in descending order, and limiting the result set to the specified number of rows.

`[n]`	You must specify an integer. There is no implicit default.

Example

Example 1: Return the top ten values for a field and filter results with non-empty destination values.

```
|top 10 action

|where dest != "" | top 10 dest
```

Example 2: Returns the 10 most frequently occurring destination ports across OCSF events, ranked by highest count

```
ocsf | top 10 dst_endpoint.port
```

transaction

Use the `transaction` operator to group events together based on different grouping methods. Each group is considered as a transaction.

Syntax

```
|transaction <field list> [combolimit]
[startpattern] [endpattern] [events] [span] [pause]
[outputlimit]
```

`<field list>`	Comma-separated list of fields to define unique file combinations for the 1st-level grouping.
`[combolimit = <num>]`	(Optional) Maximum number of combinations based on the field list.
`[startpattern = <string>], [endpattern = <string>]`	(Optional) Use the specified patterns to specify the start and end of what defines a transaction. You can specify matching patterns using AQL `where` operator evaluation expressions, as shown in the following examples: • `sourcetype startswith 'ms_d'`, or • `example_field contains_ci 'CVE-'`
`[events = <num>]`	(Optional) Use event count to define what a transaction is.
`[span= [<integer>] [s,h,m]]`	(Optional) Use timespan to define what a transaction is.
`[pause= [<integer>] [s,h,m]]`	(Optional) Use time pause to separate transactions.
`[outputlimit = <num>]`	(Optional) Maximum output per the combination of specified fields.

Usage

The *1st-level grouping* enumerates all possible value combinations based on the given `<field list>` (for example, `src_ip`, `dest_ip`) with `combolimit` as the maximum number of combinations, after which output is truncated.

The *2nd-level grouping* (a transaction) will be applied based on one of the following sub-grouping methods:

`[startpattern, [endpattern]`	Define the transaction based on start/end patterns.
`[events]`	Define the transaction based on the number of consecutive events.
`[span]`	Define the transaction based on the timespan of consecutive events.
`[pause]`	Define the transaction based on maximum time pause between consecutive events.

Examples

Example 1: Group events by specified fields and create transactions with a five-minute time span.

```
|where sourcetype='ms_defender' and src is not null and src_ip is
not null | transaction src,src_ip span=5M combolimit=4
```

Example 2: Create transactions for events with matching source IP addresses and source values.

```
|where sourcetype='ms_defender' and src_ip is not null |
transaction src_ip,src events=10
```

Example 3: Groups Fortigate Firewall events into transactions by source interface and IP within five minute windows, capping each transaction at four events.

```
ocsf | where metadata.product.name='Fortigate Firewall' and src_
endpoint.interface_name is not null and src_endpoint.ip is not
null | transaction src_endpoint.interface_name, src_endpoint.ip
span=5M combolimit=4
```

Example 4: Creates an inline session dataset for two users, converts millisecond timestamps to readable times, then groups events into transactions per user

bounded by login/logout pairs, capping each session at 5 events or 120 seconds with a 30-second pause threshold.

```
ocsf

| makeresults format='csv' input='event_meta_id,event_time_
ms,user_name,activity_name,src_
ip,dest\n1,1677485890000,userA,login,10.0.0.1,app\n2,167748590500
0,userA,view,10.0.0.1,app\n3,1677485920000,userA,logout,10.0.0.1,
app\n4,1677485935000,userA,view,10.0.0.1,app\n5,1677485950000,use
rA,login,10.0.0.1,app\n6,1677485965000,userA,view,10.0.0.1,app\n7
,1677485980000,userA,logout,10.0.0.1,app\n8,1677485995000,userA,v
iew,10.0.0.1,app\n9,1677485890000,userB,view,10.0.0.2,app'

| calc event_time=from_unixtime(event_time_ms)

| transaction user_name startpattern=(activity_name='login')
endpattern=(activity_name='logout') events=5 span=120s pause=30s
format='transactions'
```

transpose

Use the `transpose` operator to create a pivot table for further use in visualizations, similar to a pivot table in MS Excel.

Syntax

```
|transpose row=<first_column_field> col=<column_of_
field_values>
```

`first_column_field`	Required. Specify the field to be displayed as the first column in the resulting table.
`column_of_field_values`	Required. Specify the field whose values will be used as columns in the resulting table. The resulting table supports up to 10 values for columns. If there are more than 10 values, then the rest of them will be grouped in a separate column called `others`.

Usage

You can use the `transpose` operator to create a cross table that can be displayed in any chart that has two axes.

The values related to the first numeric field, different to the fields specified in `row=<field1>` and `col=<field2>`, will be displayed as values in the table.

Examples

Example 1: Transpose rows to columns using state as row field and category as column field.

```
|transpose row=state col=category
```

Example 2: Transpose rows to columns using year as row field and category as column field.

```
|transpose row=year col=category
```

Example 3: Transpose rows to columns using status as row field and year as column field.

```
|transpose row=status col=year
```

Example 4: Buckets OCSF events hourly, counts per activity type, then pivots so each activity becomes a column and each row represents an hour.

```
ocsf | bin event_time span=1h | aggr count by event_time,
activity_name | transpose row=event_time col=activity_name
```

turbosearch

Use the `turbosearch` operator for:

- Forensic and retrospective searches on large datasets
- Threat hunting
- Needle in a haystack scenarios

When Schema RBAC is active for your organization, the schema filters in your assigned roles determine which data this feature returns. An eventlog filter limits your results to only the eventlog data your role permits, and an OCSF filter limits your results to only the OCSF data your role permits. If a query runs against a schema for which your role has no filter assigned, the query returns a forbidden error. For details, see "Role-Based Access Control for Schemas" on page 110.

Syntax

> **Note:** `retrosearch` is an alias of `turbosearch` and has the same behavior.

keyword-based:

```
|turbosearch "<value>"
|turbosearch "<value>" [, "<value1>", "<value2>",...]
```

field-based:

```
|turbosearch <key>=<value>
|turbosearch <key>=<value> [, <key1>="<value1>",
<key2>="<value2>",...]
```

lookup table-based:

```
|turbosearch [<your_lookup_table> | fields <your_field_name>]
```

Usage

You can use the purpose-built `turbosearch` operator to issue a retrospective search, that includes both keyword- and field-based search queries.

You can also combine this operator with all other search operators, including support for logical operators AND, OR, and IN. See "Examples" on page 694 for more information.

> **Note:** When you use the `turbosearch` operator with other operators, you must specify it as the first operator in the search query. All other operators must be chronologically placed after

```
turbosearch.
```

"<value>"	Specify the value on which you would like to issue a retrospective search, enclosed either in single or double quotes. The argument is not case sensitive. Currently, you can search on the values for the following indicator types: • IP Addresses: [src_ip, dest_ip, src, dest] • Domains: [src, dest, query] • File hashes: [file_hash, file_hash2, file_hash3] • Email addresses: [sender, receiver] • URLs: [url, http_referrer] turbosearch also supports searches on the following fields: • user • host • file_name • account_id • dvc_name • message_id If the list of IOCs exceeds 20 indicators, turbosearch will automatically turn this search into a background job. See Job Tracking to understand how you can monitor the status of this background job. **Note:** Out of the box, turbosearch supports the fields listed above.

	However, if you require additional fields from the eventlog schema, contact Anomali Support to enable those for your need.
`[, "<value1>", "<value2>",...,"<valueN>"]`	Optional. Specify one or more values on which to issue a search. Enclose the values in single or double quotes.
`|turbosearch [<your_lookup_table> | fields <your_field_name>]`	Provide a lookup table that has one or more IOCs. Turbosearch will automatically turn this search into a background job. See Job Tracking to understand how you can monitor the status of this background job. Example: `|turbosearch [example_lookup_table | fields example_lookup_table_field]` In the example query above, turbosearch searches the eventlog data against the `example_lookup_table` lookup table, that contains indicators, and the `example_lookup_table_field` field in this lookup table, and then triggers a background job that automatically saves the result into a new lookup table.

Examples

Basic examples

Example 1: Search for a keyword by IP address.

```
|turbosearch "1.2.3.4"
```

Example 2: Search for a field-based indicator using IP address syntax.

```
|turbosearch src_ip = '1.2.3.4'
```

Example 3: Search for an email address field value.

```
|turbosearch sender='sender_example@example.com'
```

Example 4: Search for a keyword by URL value.

```
|turbosearch "https://www.example-url.com/feed"
```

When combined with other operators

Example 5: Search by IP address and filter results by specified fields.

```
|turbosearch "1.2.3.4" | fields action, src , dest | limit 100
```

Example 6: Search by IP address and hostname with multiple field criteria.

```
|turbosearch src_ip = "1.1.1.1" and host = "MX56WAB671"| fields
action, src , dest | limit 100
```

Example 7: Search by hostname or user with multiple values.

```
|turbosearch host = "CD12WRT101" or user in ("<foo1@bar1.com>",
"<foo2@bar2.com>”)
```

Example 8: Search by multiple indicators and aggregate by source type.

```
|turbosearch "tech@threatstream.com", "examplefilehash" | aggr
count (sourcetype) as srctype_count
```

Example 9: Search by URL and filter results by source type values.

```
|turbosearch 'https://www.example-url.com' | where sourcetype
contains ("aws", "cloud", "vpc")
```

Example 10: Search using a lookup table containing indicators.

```
|from table threat_indicators | turbosearch indicator
```

Using turbosearch with OCSF

The `turbosearch` operator supports OCSF queries. Prefix your query with `ocsf` and use the same `turbosearch` syntax to perform retrospective searches on OCSF event data.

Syntax:

```
ocsf | turbosearch "<value>"
```

Example: Search OCSF Network Activity events for a specific IP address.

```
ocsf | where category_name = "Network Activity" | turbosearch
"10.0.0.1"
```

When OCSF RBAC enforcement is enabled, `turbosearch` respects the OCSF filter defined in the user-assigned role. Results are automatically constrained to the OCSF categories the user has permission to access. If your account does not have an OCSF role assigned, OCSF queries return an `ocsf is forbidden` error. Contact your organization administrator to have an OCSF role assigned. See "Role-Based Access Control for Schemas" on page 110 for more information.

tsquery

Use the `tsquery` operator to search ThreatStream observable or threat model data.

Syntax

```
|tsquery type=<type> search='<search>' [aggs=<aggs>]
[histogram=<boolean>]
```

`type=<type>`	Specify a search type: • `ioc`: Search ThreatStream observables. • `threatmodel`: Search ThreatStream threat model entities.
`search=<search>`	Specify a ThreatStream search expression as you would in the ThreatStream advanced search bar. For information about ThreatStream search expressions, refer to the following ThreatStream online help topics: • Performing Advanced Observable Searches • Performing Advanced Threat Model Searches
`aggs=<aggs>`	Specify a ThreatStream observable or threat model field to aggregate.
`histogram=<boolean>`	Specify `true` to return aggregated data as time series data. If not specified, the implicit default is `false`.

Usage

You can use the `tsquery` operator to perform a ThreatStream observables search or threat model search. Returning aggregated data for time series charts is optional. If you specify the `histogram=true` option, you must also specify an `aggs` argument.

Examples

Example 1: Query indicators of compromise with active status and tags matching search criteria.

```
<lookuptable>|tsquery type=ioc search='(status = "active") AND
(tag contains test) AND created_ts>30d' aggs='itype'
```

Example 2: Query threat models of type tipreport and aggregate by feed identifier.

```
<lookuptable>|tsquery type=threatmodel search='(model_
type=tipreport)' aggs='feed_id'
```

Example 3: Query threat models with specified criteria and aggregate results.

```
<lookuptable>|tsquery type=threatmodel search='(model_type=
```

Example 4: Queries the threat intelligence IOC store for active IOCs tagged with test that were created within the last 20 days, enriching the OCSF pipeline with matching threat indicator results.

```
ocsf | tsquery type=ioc search= '(status = "active") AND (tag
contains test) AND created_ts > 20d'
```

uniq

Use the `uniq` operator to remove result rows that are exact duplicates.

Syntax

`|uniq`

Usage

You can use the `uniq` operator to simplify the results table when duplicate rows are not useful to your analysis.

> **Note:** Anomali recommends you do not run this command against a large data set.

Examples

Example 1: Identify unique values in specified fields.

```
uniq_test |fields sender, day, month |uniq
```

Example 2: Buckets OCSF events by day, counts per day, then removes consecutive duplicate rows.

```
ocsf | bin event_time span=1d | aggr count as ioc_count by event_
time | uniq
```

virustotalv3

Use the `virustotalv3` operator to enrich observables with comprehensive threat intelligence using the ThreatStream VirusTotal v3 enrichment, including detailed statistics and community insights.\

> **Note:** For more information on Google Threat Intelligence, the gti operator provides enhanced threat intelligence capabilities.

Syntax

`query | virustotalv3 <ioc_field_or_literal>`

`<ioc_field_or_literal>`	The field name containing the observable.

Usage

You can use the `virustotalv3` operator to enrich your observables based on the latest VirusTotal intelligence and, therefore, inform your security orchestration, automation and response actions. For example, you can use the `virustotalv3` operator to determine whether there is community consensus about an IP address in security reporting, and then filter or tag observables based on the VirusTotal verdicts and the community consensus. This is especially useful for security analysts who need consolidated verdicts, enrichment data, and community feedback for threat detection, investigation, and automation workflows.

Specifically, the operator returns results that contain information about the primary IOC, its associated IOCs, the number of engines that flagged the observable as either being malicious, suspicious, or harmless, along with community statistics or reports assigning it one of these statuses, and the timestamp of the most recent analysis performed on the IOC.

> **Note:** The virustotalv3 operator uses the VirusTotal API key associated with the VirusTotal enrichment configured for your ThreatStream organization. If your organization uses a public API key, VirusTotal limits the number of API requests per minute and per day. Be sure to refine your query and use the `limit` operator to reduce the number of requests to VirusTotal.

Examples

Example 1: Query VirusTotal v3 API for IP address threat intelligence.

```
example_api_proxy | calc srcip= '123.122.221.231' | virustotalv3
srcip
```

where

Use the `where` operator to filter results.

Syntax

> **Note:** AQL boolean arguments are case-insensitive, that is, all the following boolean arguments are valid:
>
> **Positive case**: `True`, `true`, `t`
>
> **Negative case**: `False`, `false`, `f`

```
|where <Boolean expression>
```

Typically, the `where` operator specifies a logical expression to evaluate field-value matching:

```
|where <field> <operator> <value>
```

```
|where [<field>] [earliest|latest]=[period],
[earliest|latest]=[period]
|where [<field>] [earliest|latest]=[period] AND
[earliest|latest]=[period]
|where timerange([period], [period], [field])
|where isnotnull(<field>)
|where isnull(<field>)
|where isnum(
|where isoutlier(<field>, [param], [uselower])
|where like(<field>, <pattern>)
|where match(<field>, <regex>)
|where mvfind(<field>, <regex>)
```

For subsearch: `| where [<field>] | [your_subsearch]`

`<field>`	• Anomali event search starts with an implicit `select` from the eventLog table. • Refer to the Anomali Security Analytics Common Event Schema for descriptions of fields and example values.

`<operator>`	• You can use the following operators to match event fields with string values: `!=`, `=`, `==`, `contains`, `contains_ci`, `not contains`, `startswith`, `not startswith`, `endswith`.
	  ▪ Use `contains_ci` to perform case insensitive matching for partial strings. If you need to match an event field with any value, you can use an asterisk.
	  The following syntax is supported:
	  `|where contains_ci (*)`
	  Example:
	  `|where sourcetype contains_ci (*)`
	  ▪ Use `contains`, `contains_ci`, `startswith`, and `endswith` to match an event field with multiple string values.
	  The following syntax is supported:
	  `|where contains ('<string1>', '<string2>', ..., '<stringN>')`
	  Example:
	  `|where sourcetype contains ('aws', 'carbon', 'for', 'vpc')`
	• You can use the `REGEXP_LIKE` operator to match a regular expression. Example:
	  `|where regexp_like(sourcetype, '^ts_*')`
	• You can use the following operators to match event fields with date or numeric values (bytes_in, bytes_out, event_time, dest_port, src_port, and timestamp): `>=`, `<=`, `>`, `<`, `!=`, `=`, `==`.
	• You can use the Boolean operator `in` to check if one of the values in the list matches a value that you specify.
	  The following syntax is supported:
	  `|where in(<value>, [<list>])`
	  `|where <value> in([<field list>])`

	Example:
	`|where status in("400", "401", "403", "404")`
	`|where "203.0.113.255" in(ipaddress, clientip)`
	• You can use the `insubnet` operator to match IPv4 addresses that belong to the specified IPv4 subnet. For example:
	`|where not src contains ":" |where src insubnet "192.168.0.0/16"`
	The `insubnet` operator does not support IPv6 subnets.
	• You can use the `is null` operator to show results having null values and the `is not null` operator to exclude results having null values.

`<value>`	The event log value.
`[your_subsearch]`	The subsearch embedded within the primary search, inside square brackets (`[]`). See Subsearch documentation for more information.

`[earliest|latest]`	Specify a time range expression in the AQL query instead of passing the time range selected with the Search user interface.
	
	• `<field>`: Typically, you use `earliest/latest` functions at the end of a query when you want to apply date filtering relative to the `event_time` field. If no field is specified, `event_time` is the implicit default. However, you can use `earliest/latest` functions with any date type field having Unix epoch timestamps.
	
	• `[earliest|latest]=[period]`: The functionality differs depending on which functions are set:
	
	  ▪ `earliest` only: Returns events starting at the specified period.
	
	  ▪ `latest` only: Returns events ending at the specified period.
	
	  ▪ both: Returns events in the range of the two specified periods.
	
	• `period`: Must be one of the following:
	
	  ▪ A relative time string, which can be provided without quotes.
	
	  ▪ An absolute time string, which must be enclosed in quotes.
	
	  ▪ `now` string, which can be provided without quotes.
	
	• Relative time strings:
	
	  ▪ Must be a string in the format: `< + | - ><number><time unit>@<snap_to_specifier>`
	
	  ▪ `+` denotes a future time.
	
	  ▪ `-` denotes a past time.
	
	  ▪ No sign being provided is semantically the same as providing `-`, meaning strings without a sign are assumed to be past relative times. In other words, `2d`

	is the same as `-2d`.
	■ About `snap_to_specifier`:
	◦ `@<snap_to_specifier>` is an optional specifier that rounds down to the earliest or latest time by the amount you specify, in your timezone. To do so, insert the "@" character between the time amount and the snap-to specifier.
	◦ Supported specifiers are: `@s`, `@m`, `@h`, `@d`, `@mon`, for seconds, minutes, hours, days, and months respectively.
	◦ By default, the search uses seconds as the snap time unit.
	Absolute time strings must be in the format: `"%Y-%m-%d %H:%M:%S"`
	`now` string: Specifies the current time (the time at which the search is being executed).
	Examples:
	`|where event_time earliest=1d, latest=now --Get all events from the range of 1d back up to now`
	`|where event_time earliest=-1d --Get all events prior to 1d back from now`
	`|where event_time earliest='2023-11-05 12:00:00' --Get all events starting at: '2023-11-05 12:00:00'`
	`|where event_time latest=1d --Get all events ending at 1d back from now`
	`|where event_time latest='2023-11-05 12:00:00'--Get all events ending at: '2023-11-05 12:00:00'`
	`|where event_time earliest=-2d@d --On April 28th, the search looks for events starting from 12:00 AM on April 26th`
	`|where event_time earliest=-30d@mon, latest=@mon --Get all events from the start of the previous month to the end of that month. For example, running this on Nov 1st will look for events between Oct 1st and Oct 31st.`
	Subsearch:
	You can define a time range for subsearch that is different from the range for the primary search. See Subsearch

	documentation for more information. `| aggr count by sourcetype | appendtable [| where earliest=10d and latest=5d | aggr count by sourcetype]`

`|where timerange ([period], [period], [field])`	Specify a time range expression in the AQL query instead of passing the time range selected with the Search user interface. • `[period], [period]`: At least one `period` argument must be specified. The first `period` refers to the start point of the time range; the second `period` refers to the end point. If only one `period` argument is specified, it is assumed to be the start point and the current time is the end point. `period` can be specified in the following formats: ▪ A relative time string, which can be provided without quotes. ▪ An absolute time string, which must be enclosed in quotes. ▪ `now` string, which can be provided without quotes. Relative time strings: ▪ Must be a string in the format: `< + | - ><number><time unit>` ▪ `+` denotes a future time. ▪ `-` denotes a past time. ▪ No sign being provided is semantically the same as providing `-`, meaning strings without a sign are assumed to be past relative times. In other words, `2d` is the same as `-2d`. Absolute time strings must be in the format: `"%Y-%m-%d %H:%M:%S"` `now` string: Specifies the current time (the time at which the search is being executed). • `[field]`: Typically, you use the `timerange` function at the end of a query when you want to apply date filtering relative to the `event_time` field. If no field is specified, `event_time` is the implicit default. However, you can use

	the `timereange` function with any date type field having Unix epoch timestamps.
	
	Examples:
	`|where timerange('1d', 'now', event_time) --Get all events from the range of 1d back up to now`
	`|where timerange('1d') -- Functionally the same as the above example, 'now' is assumed for the second period, event_time is assumed for the field`
	`|where timerange ('2023-11-05 12:00:00') -- Get all events from '2023-11-05 12:00:00' up to now`
	`|where timerange ('2023-11-05 12:00:00', '1d') -- Get all events from '2023-11-05 12:00:00' up to 1d back from the current time the search is executed at`

`isnotnull(<field>)` `isnull(<field>)`	Filter results to exclude logs having null values for the specified field; or to return only logs having null values for the specified field. Example: `|where isnotnull(src)|fields sourcetype, src`
`isnum(<field>)`	Filter results to include only logs having number values for the specified field. Example: `|where isnum(user)`

`isoutlier(<field>, [param], [uselower])`	Filter results to include those where values for the specified field are outliers.
	Optional:
	• `param`: Specify a floating point value. This parameter multiplies the interquartile range (IQR) in the evaluation. Increasing the value yields fewer outliers; decreasing the value yields a larger number of outliers. If this parameter is not specified, the implicit default is 2.5.
	• `uselower`: Specify `true` to detect outliers below the median. If this parameter is not specified, the implicit default is `false`.
	Examples:
	`|where isoutlier(bytes_in)`
	`|where isoutlier(bytes_out, true)`

`like(<field>, <pattern>)`
`not like(<field>, <pattern>)`
`<field> like <pattern>`
`<field> not like <pattern>`
	Filter results to include only logs having values for the specified field that match the specified pattern. The pattern can specify an exact match or a wildcard match:
	• Use the percent (%) symbol as a wildcard for matching multiple characters.
	• Use the underscore (_) character as a wildcard to match a single character.
	Refer to the Examples section below.

`|where match (<field>, <regex>)`	Filter results to include only logs having values for the specified field that match the specified regex. Example: `|where match(event_name, 'Hello W..ld')`
`|where mvfind (<field>, <regex>)`	Filter results to include only logs having values that match the `mvfind()` evaluation. The `mvfind()` function returns the index for the first value in a multivalue field that matches a regular expression. The index begins with zero. If no values match, returns NULL. Example: `|where mvfind(ports, "8080") >= 0`

Usage

Use the `where` operator to build complex filters to focus on data that impacts your business.

Evaluation of logical operators

You can create compound filters with use of parentheses, Boolean `NOT`, Boolean `AND`, or Boolean `OR` clauses.

Search evaluates the logical operator in the following descending order (first to last evaluation):

1. Parentheses ()
2. Comparison operators (=, !=, <, >, <=, >=)
3. Logical NOT (NOT)
4. Logical AND (AND)
5. Logical OR (OR)

Example 1: Search with AND and OR

The Search query

```
| where category != '' or src = '50.1.114.34' and sourcetype =
'ts_cloud'
```

during evaluation becomes

```
| where (category != '') or (src = '50.1.114.34' and sourcetype =
'ts_cloud')
```

This means that the condition `src = '50.1.114.34' and sourcetype = 'ts_cloud'` will be evaluated first, and then the result will be combined with `category != ''` using the OR operator.

Example 2: Search with NOT, AND and OR

The Search query

```
| where event_time NOT IN (9, 10) AND sourcetype contains_ci
('aws') or sourcetype contains_ci ('fortinet') and sourcetype =
'abc'
```

during evaluation becomes

```
| where (event_time NOT IN (9, 10) AND sourcetype contains_ci
('aws')) or (sourcetype contains_ci ('fortinet') and sourcetype =
'abc')
```

This means that the condition `event_time NOT IN (9, 10) AND sourcetype contains_ci('aws')` will be evaluated first, and then the result will be combined with `(sourcetype contains_ci ('fortinet') and sourcetype = 'abc')` using the OR operator.

Examples

Example 1: Filters events to CrowdStrike logs where the process image is pnputil.exe, a Windows utility sometimes abused for malicious driver installation.

```
|where sourcetype contains "CrowdStrike" and image endswith
"\pnputil.exe"
```

Example 2: Filters events to internal traffic originating from the 192.168.0.0/16 subnet where the destination is both non-null and non-empty — effectively guarding against incomplete or malformed network flow records.

```
|where (src insubnet '192.168.0.0/16' and dest is not null) AND
(src insubnet '192.168.0.0/16' and dest !='')
```

Example 3: Filters events to Cisco logs that have a non-null source IP.

```
|where sourcetype contains "cisco" and src_ip is not null
```

Example 4: Filters OCSF events to Fortigate products, takes the first 20, splits the product name into a multi-value field on spaces, then calculates how many words are in each product name.

```
ocsf | fields metadata.product.name | where metadata.product.name
contains 'Fortigate' | limit 20 | makemv delim = ' '
metadata.product.name | calc mv_len = array_len
(metadata.product.name_mv)
```

whois

Use the `whois` operator to enrich an IP address or field containing an IP address using the ThreatStream WHOIS and DomainTools enrichments configured for your organization.

Syntax

```
|whois [PARSED=<parse_whois_output>] <field or
value>
```

`<field or value>`	Specify the IOC or value to look up.

`<parse_whois_output>`	Optional. Specify whether to parse the raw WHOIS data into individual structured fields. Possible values:
	• `true`: Parses the raw WHOIS response. By default, returns `whois_data` and `whois_domain_tools`. Additionally, returns new columns:
	  ▪ `whois_domain_name`
	  ▪ `whois_creation_date`
	  ▪ `whois_domain_status`
	  ▪ `whois_name_servers`
	  ▪ `whois_registrant_name`
	• `false`: returns only the raw `whois_data` and `whois_domain_tools` columns.
	Default: `false`.

Usage

You can use the `whois` operator to retrieve WHOIS enrichment details for an IP address or domain. By default, the operator returns the raw `whois_data` and `whois_domain_tools` columns.

When you set `PARSED=true`, the operator adds the following columns to the results:

- `whois_domain_name`: Registered domain name.
- `whois_creation_date`: Date the domain was registered.
- `whois_domain_status`: Current status of the domain, for example, `active`.
- `whois_name_servers`: Name servers associated with the domain.
- `whois_registrant_name`: Name of the domain registrant.

These new columns let you filter, aggregate, and query WHOIS data in downstream queries without manually parsing the raw JSON.

Examples

Example 1: Query WHOIS information for a domain name, enclosed in double quotes.

```
|whois "www.anomali.com"
```

Example 2: Query WHOIS information for an IP address.

```
|whois "192.0.2.1"
```

Example 3: Query WHOIS information for an IP address based on a custom user-defined variable.

```
|calc ioc_value = "192.0.2.1" |whois ioc_value
```

Example 4: Query WHOIS information for indicators filtered by type.

```
iocmatch | where ioc_type = 'domain' | limit 10 | whois indicator
```

Example 5: Retrieves parsed WHOIS data for an IP address and returns domain name, registrant name, domain server name, domain status, and creation date in structured columns alongside the raw WHOIS response.

```
|whois PARSED=true "8.8.8.8"
```

Example 6: Retrieves WHOIS registration data for `www.google.com`.

```
ocsf | whois "www.google.com"
```

xmlkv

Use the `xmlkv` operator to extract key-value data from fields having key-value data in XML format.

Syntax

```
|xmlkv [<field>] [maxinputs=<int>]
```

`[<field>]`	Specify a field having key-value data in XML format. If field is not specified, search attempts to parse the `event._raw` field if it contains an XML string.
`[maxinputs=<int>]`	Maximum rows of results that can be parsed by the operator per invocation. If a query returns more than the maximum, the operator is invoked multiple times until all results are processed. The default is 5000. In general, do not specify maxinputs unless you encounter issues getting results using the default.

Usage

You can use the `xmlkv` operator when the key-value data in the XML field is of interest.

Examples

Example 1: Extract key-value pairs from XML data.

```
|xmlkv xmlfield
```

Example 2: Extract key-value pairs from XML with maximum input limit.

```
|xmlkv maxinputs=10000
```

xmlunescape

Use the `xmlunescape` operator to unescape XML specific sequences from a specified field.

Syntax

```
|xmlunescape <field>
```

`<field>`	Specify a field having XML data.

Usage

You can use the `xmlunescape` operator when you need to unescape XML data.

The XML mapping is as follows.

```
xml_mapping = {
   '&quot;': '"',
   '&apos;': "'",
   '&lt;': '<',
   '&gt;': '>',
   '&amp;': '&'
}
```

Examples

Compare results for the following two example queries to see how the `xmlunescape` operator transforms the value for the column `xmlnum`:

Example 1: Creates a field containing an HTML-escaped XML string and returns it as-is.

```
|calc xmllum="&lt;test&gt;test&lt;/test&gt;" | fields xmllum
```

Example 2: Creates the same escaped XML string then unescapes it back to valid XML

```
|calc xmllum="&lt;test&gt;test&lt;/test&gt;" |fields xmllum
|xmlunescape xmllum
```

Example 3: Creates the same escaped XML string sourced from OCSF events, then unescapes it back to valid XML

```
"ocsf | calc xmllum='&lt;test&gt;test&lt;/test&gt;' | fields
xmllum | xmlunescape xmllum"
```

xpath

Use the `xpath` operator to extract the attribute specified by an XPath expression from the specified field.

Syntax

```
|xpath <xpath-string> input=<field> [output=<field>]
[default=<string>]
```

`<xpath-string>`	Specify an XPath reference to the attribute to be extracted. The syntax described in the Python Standard Library 19.7.2.2. Supported XPath syntax is supported.
`input=<field>`	Specify the field to find and extract the referenced XPath value from.
`[output=<field>]`	Specify a new field name for the extracted data. The default if this option is not specified is `xpath`.
`[default=<string>]`	Specify a value to write to the output field if no value exists at the location specified by the XPath expression. If this option is not defined, no default value is written.

Usage

You can use the `xpath` operator to extract values from a single element in `_raw` XML event fields.

For example, a `_raw` XML event field has data similar to the following:

```
<foo>
   <bar nickname="spock">
   </bar>
</foo>
```

```
<foo>
    <bar nickname="scotty">
    </bar>
 </foo>
 <foo>
    <bar nickname="bones">
    </bar>
 </foo>
```

You can use the `xpath` operator to extract the nicknames spock, scotty, and bones and output those values to the `name` field. Refer to the example below.

Example

Example 1: Extract XML attributes using XPath expression.

```
|xpath  "//bar/@nickname" input=myfield output=na
```

xyseries

Use the `xyseries` operator to convert a set of results into a format suitable for creating a graph.

Syntax

`|xyseries <x_field> <y_name> <y_data>`

`<x_field>`	Specify the field that represents the x-axis of the graph.
`<y_name>`	Specify the field to populate the y-axis of the graph.
`<y_data>`	Specify the field or list of fields that contains data for the graph.

Usage

You can use the `xyseries` operator to plot a graph to detect anomalies in web traffic, such as anomalies in page views at specific times or changes in response times at specific times.

Examples

Example 1: Create series data with source IP and URL as dimensions.

```
|xyseries src_ip url return_code
```

yieldtable

Use the `yieldtable` operator to generate a lookup table having the name you specify and the columns and rows from the results generated when the search is run.

Syntax

```
|yieldtable <table name> [is_temp={false|true}]
[append={false|true}]
```

```
|outputcsv <table name> [is_temp={false|true}]
[append={false|true}]
```

`<table name>`	Specify a table name. If you specify the name of an existing table, that table is overwritten with the results from your query.
`is_temp={false|true}`	Optional. Specify `true` to allow the lookup table to be deleted from temporary storage after the TTL expires. Specifying `true` makes it a “temp table”. Specify `false` to set no TTL for the table. In effect, this setting creates a more permanent lookup table. See Managing Lookup Tables for information about lookup tables.
`append={false|true}`	Optional. Specifying `true` will append the results to the existing lookup table. Specifying `false` will overwrite the old table and create a new lookup table with the generated results.

Usage

The `yieldtable` operator should be the last piped operator in a query. It is used to publish the results of the query formed by the piped expressions preceding it into a temporary results table. Anomali recommends you refine the results table using `where` filters and `fields` lists to produce refined lookup tables that meet your requirements.

`yieldtable` supports OCSF nested object fields when creating lookup tables. The lookup table schema preserves the top-level object fields, for example, `src_endpoint`, along with the full nested field structure.

> **Note:** Lookup tables created using the `yieldtable` operator cannot be updated.

The `outputcsv` operator is an alias of the `yieldtable` operator. In other words, the usage for `outputcsv` is exactly the same as the usage for `yieldtable`.

Examples

Example 1: Save top ten users to a new lookup table.

```
|top 10 user |yieldtable top_10_user_tabl
```

Example 2: Counts OCSF events per user per hour, then writes the result to a temporary table `ocsf_qa_temp_table_append_false`, overwriting any existing data.

```
ocsf | timetrend count as event_count by actor.user.name span=1h
| yieldtable ocsf_qa_temp_table_append_false is_temp=true
format=parquet append=false
```

Example 3: Create a lookup table from OCSF events using OBJECT sub-fields, then query the resulting table.

```
ocsf
```

```
                | where src_endpoint.ip != ''

                | fields event_time, src_endpoint, dst_endpoint

            | yieldtable ocsf_endpoint_lookup_example is_
temp=true format=parquet append=true
```

Then, to query from the resulting lookup table:

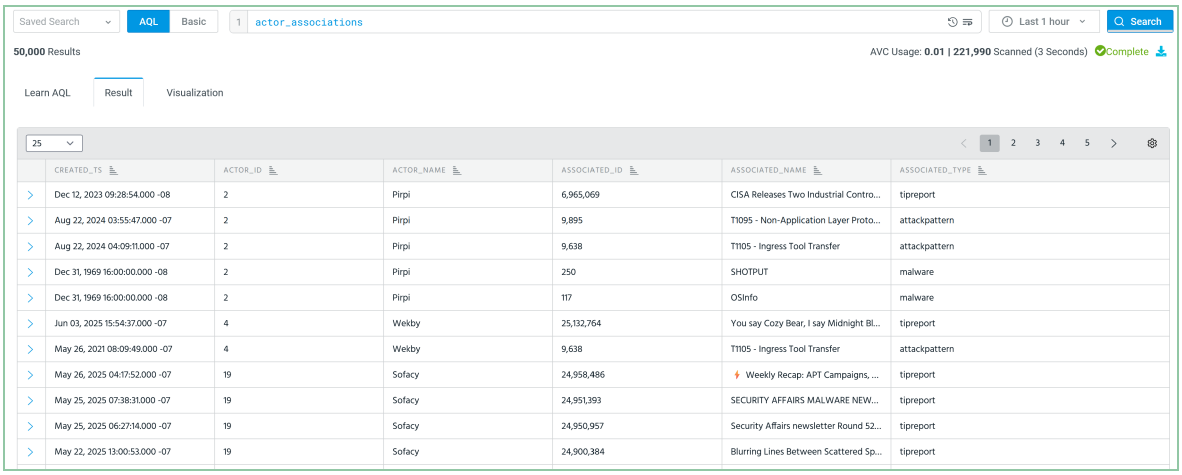
```
ocsf_endpoint_lookup_example | iplocation src_endpoint.ip |
fields src_endpoint.ip, country, lat, lon
```

Reference Information

This section contains additional information on different aspects of AQL Search.

actor_associations

The `actor_associations` lookup table is one of the data sources for the AI Actor Profile dashboard. It includes information about threat actor associations available in your ThreatStream instance. The table is read-only and not subject to modification.



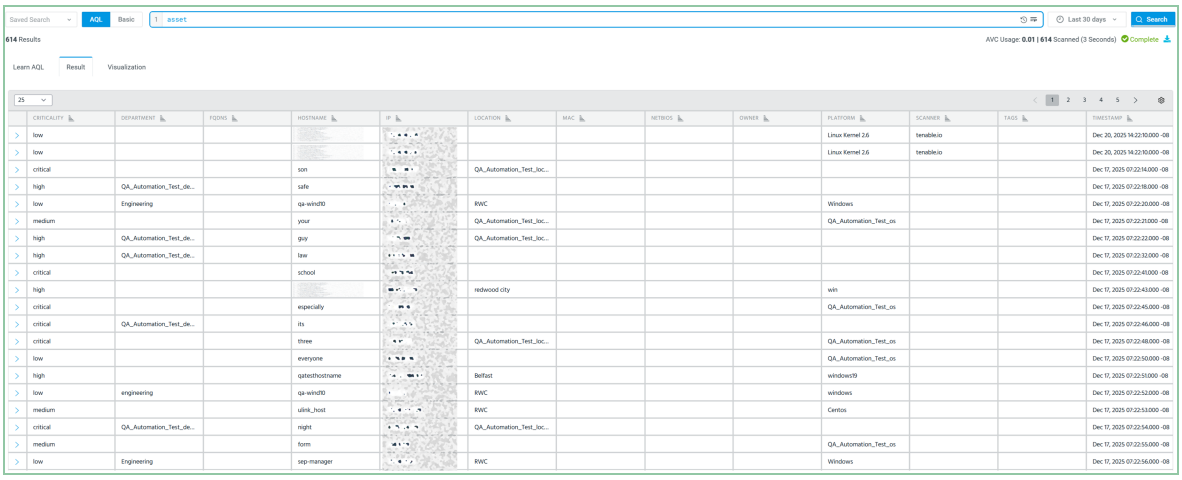
Field	Value Type	Description	Examples
actor_id	string	ID of the Actor.	• 5 • 22 • 55
actor_name	string	Name of the Actor.	• Pirpi • Weekby • Axiom

Field	Value Type	Description	Examples
associated_id	string	ID of the association for the Actor.	• 204 • 26,994,362 • 9,895
associated_name	string	Name of the association for the Actor.	• SHOTPUT • HTTPBrowser • T1105 - Ingress Tool Transfer • c487c830ff5a342513603a5cb75bd6de
associated_type	string	Threat model type of the association for the Actor.	• malware • tipreport • attackpattern

Field	Value Type	Description	Examples
created_ts	date	Time stamp of when the Actor was created on ThreatStream. Date is specified in the following format: `MMM dd, yyyy HH:mm:ss.SSS Z` where • `MMM` - month name • `dd` - day of the month • `yyyy` - year • `HH` - hour (24-clock, 2 digits) • `mm` - minutes • `ss` - seconds • `.SSS` - milliseconds (3 digits) • `Z` - timezone offset from UTC	Aug 22, 2024 03:55:47.000 -07

asset

The `asset` lookup table stores metadata of imported assets such as hostnames, IPs, ownership, criticality, FQDNs, operating systems, or location, among several more. This table is automatically refreshed whenever new asset data is ingested. This asset data enriches search results with asset context for more accurate monitoring and reporting. The table is read-only and cannot be manually modified.



Field	Value Type	Description	Example
asset_id	string	Uniquely identifies the asset across data sources and imports.	device-12345
asset_kind	string	Defines the high-level class of the asset.	device
asset_type	string	Specifies the functional type of the asset.	server

Field	Value Type	Description	Example
asset_category	string	Categorizes the asset based on its role or usage context.	endpoint
asset_risk_score	int	Represents a computed risk score for the asset based on security posture and exposure.	78
asset_role_function	string	Used to determine whether observed activity aligns with normal role-based behavior.	firewall, dmz, database
asset_status	string	Indicates the current operational or lifecycle state of the asset.	Active
alt_fqdns	array	Lists alternate fully qualified domain names associated with the asset.	["webserver01.internal.example.com"]

Field	Value Type	Description	Example
alt_ip_ addresses	array	Contains additional IP addresses observed for the asset.	["10.0.1.7"]
alt_mac_ addresses	array	Stores alternate MAC addresses linked to the asset.	["00:1b:63:84:45:e7"]
confidence	int	Indicates the confidence level of asset identification and correlation.	95
criticality	string	Security importance of the asset.	low, medium, high, critical
department	string	Identifies the organizational unit or team that owns or uses the asset.	Engineering, IT
environment	string	Describes the deployment environment in which the asset operates.	Production

Field	Value Type	Description	Example
first_seen	string	Records the timestamp when the asset was first observed.	2024-10-01T08:12:00Z
fqdns	string	Stores the fully qualified domain name associated with the asset.	qa-win10.example.com
hostname	string	Represents the system hostname used to identify the asset on the network.	webserver01
ip_address	string	Contains the IPv4 or IPv6 address assigned to the asset.	1.2.3.4
is_critical_asset	boolean	Indicates whether the asset is business-critical or security-sensitive. Used as a risk multiplier in scoring and prioritization.	true

Field	Value Type	Description	Example
last_seen	string	Records the timestamp when the asset was most recently observed.	2025-01-05T14:25:00Z
last_change_date	string	Timestamp of the last significant configuration or state change on the asset. Helps reduce noise and improve change-aware detections.	2026-01-08T22:14Z
last_patch_date	string	Timestamp of the most recent successful patch applied to the asset. Helps correlate behavioral changes and adjust risk scoring.	2026-01-02T03:15Z
location	string	Describes the physical or logical location of the asset.	Data Center 1

Field	Value Type	Description	Example
mac_ address	string	Stores the MAC address of the network interface associated with the asset.	00:1b:63:84:45:e6
netbios	string	Captures the NetBIOS name of the asset, if available.	QA-WIN10
network_ zone	string	Identifies the network trust zone where the asset resides. Provides directional context and helps determine asset exposure and criticality.	prod
owner	string	Identifies the individual or group responsible for the asset.	IT Ops, Security Team

Field	Value Type	Description	Example
patching_ state	string	Represents the current patch lifecycle status of the asset (normal, actively patching, or delayed). Used to dynamically tune UEBA sensitivity.	normal, in_progress, extended
peer_group	string	Logical grouping of similar assets (by role, platform, and environment) used for peer-based baselines and comparative risk scoring.	linux_servers_prod
platform	string	Specifies the operating system or platform running on the asset.	Windows, Cent OS, Linux

Field	Value Type	Description	Example
platform_version	string	Specifies the version of the operating system or platform running on the asset.	20.04
scanner	string	Indicates the source or tool that imported or discovered the asset data.	tenable.io
source	string	Identifies the data source that provided or last updated the asset record.	EDR
tags	array	Holds user-defined or system-generated labels for classification or filtering.	["pci", "production", "qa"]
timestamp	long (Epoch milliseconds)	Records the time when the asset record was last updated or ingested.	1734682920000

Field	Value Type	Description	Example
vulnerabilities	array	Stores the number of known vulnerabilities associated with the asset, typically sourced from the scanner or import feed.	[{ "cve_id": "CVE-2024-12345", "severity": "high", "status": "open" }]

investigation

The `investigation` lookup table stores metadata for security investigations created in Anomali ThreatStream. Each record represents a single investigation used to track and manage threat analysis workflows, including assignment, priority, status, and associated context. The table is read-only, cannot be modified, and is automatically updated as investigations are created or modified.

Field	Value Type	Description	Example
assignee	string	The user or workgroup currently assigned to work on the investigation. Supports both individual users and team workgroups.	john.doe@acme.com
created_ts	datetime	Timestamp when the investigation record was first created. Set automatically on creation and never changes.	Apr 10, 2026 02:19:31.849

Field	Value Type	Description	Example
description	string	Long-form text describing the scope of the investigation, findings, or context. Indexed for full-text search.	Structured format for threat actor profiling.
event_time	datetime	Timestamp associated with the underlying event or activity that triggered or is linked to the investigation.	Apr 10, 2026 02:19:33.000
id	int	Auto-incrementing primary key that uniquely identifies the investigation.	78370
modified_ts	datetime	Timestamp of the most recent update to the investigation record. Automatically updated on any change.	Apr 10, 2026 02:19:33.032
name	string	Title or name of the investigation. Required and unique within an organization.	Test-Investigation-0812467890
priority	string	Urgency level assigned to the investigation to guide analyst triage and response.	very low, low, medium, high, very high
reporter	string	The user who originally created or reported the investigation.	jane@acme.com
source_type	string	Indicates how the investigation was originally created.	user, rules, phishing, import, submission, resilient

Field	Value Type	Description	Example
submitter	string	The user who submitted the investigation, which may differ from the reporter when submissions are automated or delegated.	jane@acme.com
status	string	Current workflow state of the investigation.	unassigned, in-progress, pending, completed
tags	array	User-defined or system-generated text labels for categorizing and filtering investigations.	["origin: file:///var/lib/jenkins", "Created by Anomali"]
tlp	string	Traffic Light Protocol designation indicating the sensitivity and sharing scope of the investigation.	white, clear, green, amber, amber+strict, red

investigation_elements

The Investigation Elements table contains elements that have been added to investigations, enriched with the full entity data from the underlying threat model or observable. Each row represents a single investigation element joined with its referenced entity, for example, an actor, malware, or vulnerability. The table is read-only and cannot be manually modified.

Field	Value Type	Description	Example
id	integer	Unique ID of the investigation element.	42
tlp	integer	Numeric TLP level assigned to the investigation element.	2

Field	Value Type	Description	Example
score	integer	Score assigned to the investigation element.	85
investigation_id	integer	ID of the investigation to which this element belongs.	117
investigation_name	string	Name of the investigation to which this element belongs.	Q4 Phishing Campaign Analysis
entity_id	bigint	ID of the referenced entity.	9591
entity_type	string	Data type of the referenced entity. Possible values: `ip`, `domain`, `email`, `md5`, `url`.	ip
entity_value	string	Value of the referenced entity.	52.233.72.222
entity_ip	string	IP address associated with the entity.	52.233.72.222
entity_rdns	string	Reverse DNS name associated with the IP of the entity.	mail.example.com
entity_owner_organization_id	bigint	ID of the organization that owns the entity.	2
entity_feed_id	bigint	Numeric ID of the threat feed that provided the entity.	145

Field	Value Type	Description	Example
entity_country	string	Two-letter ISO country code for the IP associated with the entity.	US
entity_latitude	double	Latitude of the geo location associated with the entity IP.	12.975
entity_longitude	double	Longitude of the geo location associated with the entity IP.	-80.193
entity_org	string	Organization name associated with the entity IP.	T-Mobile USA
entity_asn	string	Autonomous System Number for the IP associated with the entity.	37963
entity_status	string	Current state of the entity. Possible values: `active`, `inactive`, `falsepos`, `ongoing`, `historic`, `future`.	active
entity_is_public	string	Whether the entity is publicly visible.	true
entity_confidence	integer	ThreatStream-assigned confidence score of the entity being malicious (0 – 100).	75

Field	Value Type	Description	Example
entity_tags	string	Tags associated with the entity.	apt, phishing
entity_meta	string	Additional metadata associated with the entity.	
entity_import_session_id	bigint	ID of the import session that created the entity.	20145
entity_created_ts	timestamp	UTC timestamp of when the entity was created in ThreatStream.	2024-10-02T20:44:35
entity_modified_ts	timestamp	UTC timestamp of when the entity was last updated in ThreatStream.	2025-01-15T08:30:00
entity_itype	string	ThreatStream indicator type. Possible values include: `c2_ip`, `compromised_email`, `apt_md5`, `mal_md5`.	c2_ip
entity_source_reported_confidence	integer	Confidence score as reported by the original source (0 – 100).	82
entity_retina_confidence	integer	Confidence score calculated by ThreatStream ML scoring engine. `-1` indicates a score could not be calculated.	67

Field	Value Type	Description	Example
entity_update_id	bigint	Update identifier for the entity record.	5012
entity_expiration_ts	timestamp	Timestamp of when the entity intelligence expires.	2025-06-01T00:00:00
entity_uuid	string	Universally unique identifier for the entity.	550e8400-e29b-41d4-a716-446655440000
entity_is_anonymous	string	Whether the entity was submitted anonymously.	false
entity_tlp	string	TLP color of the entity. Possible values: `white`, `clear`, `green`, `amber+strict`, `red`.	green
entity_import_source	string	Original source of the entity. Only displayed for entities manually imported by your organization.	manual_import
entity_created_by	string	Email address of the user who created the entity.	analyst@example.com

Field	Value Type	Description	Example
entity_subtype	string	Further categorization of the entity. Values vary by entity type (for example, md5 or sha1 or sha256 for hash observables backdoor or ransomware/trojan for malware).	ransomware
entity_source_created	timestamp	Timestamp of when the entity was created by its original source.	2024-09-15T12:00:00
entity_source_modified	timestamp	Timestamp of when the entity was last updated by its original source.	2024-12-01T09:30:00
entity_name	string	Name of the entity.	T1027.001 - Obfuscated Files or Information: Binary Padding
entity_description	string	Description text of the entity.	Remote access trojan used in targeted campaigns.
entity_avatar_s3_url	string	S3 URL of the entity avatar image.	
entity_organization_id	integer	ID of the organization that owns the entity.	2
entity_is_team	string	Whether the entity is associated with a team.	false

Field	Value Type	Description	Example
entity_soph_type_id	integer	Numeric ID representing the sophistication level of the entity.	3
entity_soph_desc	string	Description of the sophistication level. Possible values: `Novice`, `Innovator`, `Expert`, `Practitioner`, `Aspirant`.	Expert
entity_start_date	timestamp	Start date of the entity.	2024-01-01T00:00:00
entity_publication_status	string	Publication status of the entity.	published
entity_is_cloneable	string	Whether the entity can be cloned.	true
entity_parent_id	integer	ID of the parent entity, if this entity is a child or sub-entity.	9591
entity_body_content_type	string	Content type of the entity body.	richtext
entity_published_ts	timestamp	Timestamp of when the entity was published.	2024-11-01T10:00:00
entity_owner_user_id	integer	ID of the ThreatStream user that created the entity.	305

Field	Value Type	Description	Example
entity_assignee_user_id	integer	ID of the user assigned to the entity.	410
entity_tags_v2	string	JSON array of structured tags associated with the entity.	["T1556.009", "phishing"]
entity_embedded_content_url	string	URL of embedded content.	
entity_embedded_content_type	string	Type of embedded content.	
entity_threat_actor_types	string	Categorization of the entity operational profile.	Nation-state
entity_roles	string	Roles associated with the entity.	
entity_goals	string	Goals of the threat actor.	
entity_resource_level	string	Resource level of the threat actor.	
entity_primary_motivation	string	Primary motivation behind the entity.	financial-gain
entity_secondary_motivations	string	Additional goals of the entity.	Destruction
entity_personal_motivations	string	Personal motivations of the entity.	

Field	Value Type	Description	Example
entity_extra_fields	string	Additional fields stored as JSON.	
entity_is_mitre	string	Whether the entity is sourced from MITRE ATT&CK.	true
entity_c_description_last_version	string	Version identifier of the last description update.	
entity_c_description_text	string	Full text description of the entity.	
entity_aliases	string	Other names by which the entity is known.	APT29, Cozy Bear
entity_killchainstages	string	Kill chain stages associated with the entity (deprecated format).	
entity_kill_chain_phases	string	Kill chain phases associated with the entity (STIX 2.1 format).	
entity_status_id	integer	Numeric ID of the entity status.	1
entity_status_desc	string	Description of the entity status.	ongoing
entity_end_date	timestamp	End date of the entity.	2024-12-15T00:00:00
entity_first_seen	timestamp	Timestamp of when the entity was first observed.	2024-06-01T00:00:00

Field	Value Type	Description	Example
entity_last_seen	timestamp	Timestamp of when the entity was last observed.	2025-01-10T00:00:00
entity_malware_types	string	Types of malware. Possible values: `adware`, `backdoor`, `bot`, `ransomware`, `trojan`, `worm`.	ransomware
entity_execution_platforms	string	OS or environments on which the malware runs.	Windows, Linux
entity_capabilities	string	Functions and features of the malware.	Keylogging, Credential theft
entity_implementation_languages	string	Programming languages used to build the malware.	C++, Python
entity_c2_protocol	string	Protocols used for C2 traffic.	HTTP, HTTPS
entity_c2_port	string	Network ports used for C2 communications.	8080, 443
entity_is_family	string	Whether the entry describes a malware family.	true
entity_owner_org_id	integer	ID of the organization that owns the entity.	2
entity_assignee_org_id	integer	ID of the organization assigned to the entity.	5

Field	Value Type	Description	Example
entity_threat_actor	string	Associated threat actor name.	APT29
entity_campaign	string	Associated campaign name.	
entity_ttp	string	Associated TTP name.	
entity_source	string	Source of the entity.	Anomali Labs
entity_original_source	string	Original upstream source of the entity.	
entity_original_source_id	string	ID of the entity in the original source system.	
entity_is_email	string	Whether the entity is email-related.	false
entity_is_category	string	Whether the entity is a category entry.	false
entity_producer	string	Producer of the entity.	
entity_cvss2_score	double	CVSS 2.0 score. Range: 0 – 10.	7.5
entity_cvss3_score	double	CVSS 3.0 score. Range: 0 – 10.	9.8
entity_is_system	string	Whether the entity is a system-generated entry.	false
entity_vendors	string	Vendors affected by the entity.	Microsoft, Cisco

eventlog

Event log fields from multiple log sources are normalized to the Anomali Security Analytics eventlog schema. See Supported Log Sources to get the complete list of supported sources.

Alternatively, Anomali Search also supports the OCSF schema. See OCSF Schema Overview for more information.

Field	Value Type	Description	Example
access_key_id	string	Identifier for the access key used to sign the request.	AKIAEXAMPLEKEYID
account_id	string	Unique identifier of the cloud provider account.	123456789012
action	string	Action captured by the event.	allowed, blocked
action_detail	string	Detail about the activity that triggered the finding in JSON format.	{"actionType": "AWS_API_CALL", "awsApiCallAction": {"api": "ConsoleLogin", "serviceName": "signin.amazonaws.com", "callerType": "RemoteIp", "remoteIpDetails": {"ipAddressV4": "192.0.2.1", "organization": {"asn": "12345", "asnOrg": "ExampleISP", "isp": "ExampleISP", "org": "ExampleOrg"}, "country": {"countryName": "United States"}, "geoLocation": {"lat": 37.7749, "lon": -122.4194}}, "userAgent": "Mozilla/5.0", "errorCode": "Failure", "awsRegion": "us-west-2"}}

Field	Value Type	Description	Example
additional_info	string	Additional information about the finding in JSON format.	{"threatListName": "KnownMaliciousIPs", "threatIntelIndicators": [{"type": "IPV4_ADDRESS", "value": "198.51.100.1", "category": "Reconnaissance", "lastObservedAt": "2023-06-08T12:34:56Z", "source": "GuardDuty"}], "sample": true}
answer	string	Answers for the DNS query.	["10.10.10.10", "10.10.10.11"]
app	string	Application detected.	Slack, Google Gmail, LinkedIn
bytes_in	long	Bytes this device or interface received.	347
bytes_out	long	Bytes this device or interface transmitted.	665
category	string	Category of alert (azure security alert).	credentialTheft, ransomware
cl_parsing	string	Internal field to track parsing issues.	{"parse_error": "JSON is corrupted"}

Field	Value Type	Description	Example
command_line	string	Complete command string of the spawned process.	C:\Windows\system32\WerFault.exe -pss -s 572 -p 3208 -ip 3208
dest	string	Destination of network traffic.	52.233.72.222 or www.google.com
dest_country	string	Specific nation or geographical region to which data, materials, or information is being directed or delivered.	us
dest_ip	string	IP address of the destination.	52.233.72.222
dest_port	integer	Destination port of network traffic.	80, 443, 8080
ds_id	string	Optional. Unique ID of a data source in a registered link.	5

Field	Value Type	Description	Example
ds_type	string	Optional. Type of a data source in a registered link.	qradar, sumo or az_sentinel, syslog
dvc_action	string	Action or operation performed by a device.	passthrough, log-only, allow
dvc_name	string	Name or identifier of a particular device or system.	FTG37D46138
dvc_type	string	Type of a particular device or system.	firewall
end_time	long	Timestamp when a network flow ended, in seconds.	1713430144
event_id	string	A Windows identification number that specifies the event type.	4624
event_name	string	Name of an event.	DnsRequest, ProcessRollup2

Field	Value Type	Description	Example
event_tags	string	Labels or identifiers used to categorize, sort, and filter events.	network,dns,proxy
evt_level	string	Classification or categorization of log messages or events based on their significance or priority.	information, notice, alert
evt_msg	string	Textual content or description associated with a long entry or event.	applications3: Multiple.Routers
evt_severity	string	Degree of impact or criticality associated with an event or log message.	low, medium, high

Field	Value Type	Description	Example
evt_type	string	Descriptive string that indicates the specific action or operation that was performed.	AwsApiCall, AwsCloudTrailInsight
ext_index	string	Splunk index name from which the event data was ingested.	fortinet
ext_source	string	Origin or path of the file or data stream from which the event data was ingested.	cribl:USEAPP1234
ext_sourcetype	string	Specific type or format of data as it was initially received or ingested.	pan:traffic

Field	Value Type	Description	Example
failed_logon_account	string	Account related to a failed logon attempt, in JSON format.	{"sid": "NULL SID", "account_name":"PCGroup", "account_domain":"ODMH-B9HRG04"}
file_hash	string	Cryptographic identifier assigned to the file object affected by the event.	ba7816bf8f01cfea414140de5dae2223b00361a396177a9cb410ff61f20015ad
file_hash2	string	Cryptographic identifier assigned to the file object affected by the event.	56857cfc709d3996f057252c16ec4656f5292802
file_hash3	string	Cryptographic identifier assigned to the file object affected by the event.	e0d123e5f316bef78bfdf5a008837577
file_name	string	Name of the file.	nodepad.exe
file_path	string	Folder containing the file.	C:\Program Files (x86)\VMware\VMware Workstation

Field	Value Type	Description	Example
host	string	Hostname or IP address where link is deployed.	172.18.16.43
http_method	string	HTTP method used in the request.	get, post, delete, patch
http_referrer	string	HTTP referrer used in the request.	https://example.com/
image	string	Executable name of the process.	svchost.exe
instance_detail	string	Detail of an Amazon EC2 instance in JSON format.	{"instanceId": "i-1234567890abcdef0", "instanceType": "t2.micro", "launchTime": "2024-05-030T12:34:56Z", "platform": "Linux", "tags": [{"key": "Name", "value": "MyEC2Instance"}]}
logon_account	string	The account used to authenticate during a logon attempt.	AS2SOASECP32E
logon_type	string	Type of logon session that occurred.	2 (interactive), 3 (network)

Field	Value Type	Description	Example
managed_customer_id	string	Unique ID of the managed customer. See Federated Search for more information. Additionally, see Managed Security Service Provider to understand more about how Anomali enables multi-tenancy.	12345

Field	Value Type	Description	Example
managed_customer_name	string	Name of the managed customer. See Federated Search for more information. Additionally, see Managed Security Service Provider to understand more about how Anomali enables multi-tenancy.	Acne Corp
message	string	Event message available when the "store raw logs" option has been configured.	1.0 2017-12-13T08:16:02.130Z Z123412341234 example.com A NOERROR UDP Region 192.168.1.1 -
message_id	string	Globally-unique message identifier.	123e4567-e89b-12d3-a456-426655440000

Field	Value Type	Description	Example
network_direction	string	Flow of data within a network.	outbound, inbound
network_info	string	A failed logon attempt originated from, in JSON format.	{"workstation_name": "ODMH-B9HRG04", "source_address":"10.191.9.76", "source_port":"5324"}
network_source	string	Network from which traffic originates, which in zscaler corresponds to the location field.	us-ny-azure
ocsf_fields	string	OCSF key-value pairs in JSON format.	{\"packets_in\":567, \"packets_out\": 345}
original_file_name	string	Original name of the file that was renamed.	test.pdf

Field	Value Type	Description	Example
packets_in	long	Packets received by a network interface.	340
packets_out	long	Packets transmitted or sent over a network interface.	347
parent_command_line	string	Complete command string of the parent process.	C:\Windows\System32\svchost.exe -k WerSvcGroup
parent_image	string	Executable name of the parent process.	notepad.exe
parent_process_id	string	Numeric identifier of the process assigned by the operating system.	4834
platform	string	Operating system of the asset.	win, mac, linux

Field	Value Type	Description	Example
policy_id	string	Identifier of the security policy rule that applied to the logged traffic.	Allow_HTTP_Traffic
policy_name	string	Name of the security policy rule that applied to the logged traffic.	23
principal_id	string	Entity (user, role, or service) that initiated the request.	AROAJKSDKFJS:sessionName
process_id	long	Numeric identifier of the process assigned by the operating system.	3683
process_info	string	Details about the process that initiated or was involved in the event.	{"New Process ID": "0x93a4", "New Process Name": "C:\\Windows\\System32\\backgroundTaskHost.exe", "Token Elevation Type": "%%1936"}

Field	Value Type	Description	Example
protocol	string	Network or email protocol.	http, tcp, udp, smtp
protocol_code	string	Code representing a protocol.	6
protocol_version	string	Version of a protocol.	IPV4
public_src_ip	string	Public IP address of the client device.	54.12.4.3
query	string	Domain to be resolved.	www.google.com
_raw	string	Raw event message.	
raw_msg_size	long	Size of a message in bytes.	137
receiver	string	Email address of the message recipient.	john.doe@amce.org
record_type	string	DNS resource record type.	A, CNAME, MX, NS, PTR, AAAA

Field	Value Type	Description	Example
reg_key	string	Registry key that the recorded action was applied to.	HKEY_LOCAL_MACHINE\SOFTWARE\RegisteredApplications
reg_value_data	string	Data of the registry value that the recorded action was applied to.	Software\Microsoft\Windows\CurrentVersion\Applets\Wordpad
reg_value_name	string	Name of the registry value that the recorded action was applied to.	Wordpad
region	string	Geographical location where data centers are located.	us-east-1, us-west-2
resource_id	string	Unique identifier of a cloud resource, which in cloud trail corresponds to the ARN field.	arn:aws:s3:::

Field	Value Type	Description	Example
resource_role	string	Role that the resource is playing in the context of the detected finding.	TARGET, SOURCE
resource_type	string	Type of a cloud resource.	AWS::S3::Bucket
return_code	string	HTTP status code or DNS response code.	200, 404, 504, NOERROR, NXDOMAIN
s3_bucket_detail	string	Detail of an S3 bucket in JSON format.	{"arn": "arn:aws:s3:::example-bucket", "name": "example-bucket", "type": "S3", "createdAt": "2024-05-30T12:34:56Z", "owner": {"id": "123456789012", "name": "example-owner"}, "tags": [{"key": "Environment", "value": "Production"}, {"key": "Project", "value": "ExampleProject"}]}
s3_path	string	Path to an object stored in Amazon S3.	s3://my-bucket/path/to/data.tar.gz
sender	string	Email address of the message sender.	bar@amce.org

Field	Value Type	Description	Example
session_id	string	Unique identifier assigned to a user's session within a software application or a network connection.	SID34534
session_issuer	string	Entity that issued the temporary security credentials, in JSON format.	{"type": "Role", "principalId": "AROAJKSDKFJS", "arn": "arn:aws:iam::123456789012:role/RoleName", "accountId": "123456789012", "userName": "RoleName"}
signature	string	Identifier for the specific signature used to detect the attack.	TP-Link.Archer.AX21.luci.stok.Command.Injection
sourcetype	string	Device vendor.	PAN Traffic, Cisco ASA, Zscaler
src	string	Source of network traffic.	192.168.134.3 or internaltest.qa.amce.org

Field	Value Type	Description	Example
src_country	string	Specific nation or geographical region from which data, materials, or information originates.	us
src_ip	string	IP address of the source.	192.168.34.3
src_port	integer	Source port of network traffic.	8080
start_time	long	Timestamp when a network flow started, in seconds.	1713430134
status	string	Status of the service.	
subject	string	Subject line of the email message.	Your daily Threat Model digest
subtype	string	Subtype of a particular device or system.	sniffer, webfilter
threat_name	string	Name of threat detected.	Trojan:Win32/Emotet

Field	Value Type	Description	Example
threat_type	string	Classification or type of threat detected.	malware, phising
title	string	Alert title (azure security alert).	
ttl	long	Time-to-live of the network packet.	3600
url	string	Fully qualified URL of the requested HTTP resource.	https://www.computerhope.com/cgi-bin/search.cgi?q=example%20search
user	string	User that requested the traffic flow.	john.dole@amce.org
user_id	string	Identifier of the user account.	356
user_agent	string	User agent used in the request.	Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:47.0) Gecko/20100101

Field	Value Type	Description	Example
user_role_arn	string	Unique identifier of IAM role, which in cloud trail corresponds to the arn field of assumedRoleUser.	arn:aws:sts::111122223333:assumed-role/Developer_Role/Session_Name
vendor_account	string	Account name associated with the vendor supplying the service.	123456789012
vendor_product	string	Product from the vendor that generated the data.	Microsoft Defender
win_evt_account_info	string	Details about the user account involved in a logged event.	{"Account Name": "LOV2P2MIMCTX168$@PRD.MKAPP.NET", "Account Domain": "PRD.MKAPP.NET", "Logon GUID": "{BDBDA233-BD14-B47F-AC70-D10B3AB82261}"}

Field	Value Type	Description	Example
win_evt_additional_info	string	Extra context or metadata about an event.	{"Ticket Options": "0x40810000","Ticket Encryption Type": "0x12","Session Encryption Type": "0x12","Failure Code": "0x0"}
win_evt_failure_info	string	Information that helps identify the cause of the failure and the context, in JSON format.	{"reason":"The user has not been granted the requested logon type at this machine.", "status":"0xc000015b"}
win_evt_group	string	Details about a security-enabled group, common in events related to group membership changes, in JSON format.	{"sid": "NULL SID", "group_name":"PCGroup", "group_domain":"ODMH-B9HRG04"}

Field	Value Type	Description	Example
win_evt_lockout	string	Event occurs when a user account is locked due to multiple failed login attempts, in JSON format.	{"sid": "NULL SID", "account_name":"PCGroup"}
win_evt_member	string	User or account that was added to or removed from a security group, in JSON format.	{"sid": "NULL SID", "account_name":"PCGroup", "account_domain":"ODMH-B9HRG04"}
win_evt_operation	string	The specific action recorded in the log.	{"Operation Type": "Object Access", "Accesses": "Unknown specific access (bit 0)", "Access Mask": "0x1", "Properties": "-"}

Field	Value Type	Description	Example
win_evt_service_info	string	Details about a service involved in a system event, particularly in security, system, and application logs.	{"Service Name": "LOV2P2MIMCTX168$", "Service ID": "PRD\\LOV2P2MIMCTX168$"}
win_evt_subject	string	Account that triggered the event, in JSON format.	{"sid": "NULL SID", "account_name":"PCGroup", "account_domain":"ODMH-B9HRG04", "logon_id": "0x64e"}
win_evt_target	string	Object or account that is the focus of the event action, in JSON format.	{"sid": "NULL SID", "account_name":"PCGroup", "account_domain":"ODMH-B9HRG04"}

OCSF Schema

Apart from the eventlog schema, event log fields from multiple log sources are also normalized to the OCSF schema.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Anomali Search supports the following OCSF Version 1.2 classes:

- "OCSF System Activity Schema" on page 1163 - Events related to system-level activities such as process creation, file operations, and kernel operations.
- "OCSF Findings Schema" on page 886 - Security findings, detections, and alerts from security tools.
- "OCSF Identity & Access Management Schema" on page 994 - Authentication, authorization, and identity-related events.
- "OCSF Network Activity Schema" on page 1062 - Network connections, traffic, and protocol-level events.
- "OCSF Discovery Schema" on page 833 - Device and service discovery events, asset inventory.
- "OCSF Application Activity Schema" below - Application-level events including web requests, API calls, and application-specific operations.

See "OCSF Schema Overview" on page 39 for more information about using OCSF with Anomali Search.

OCSF Application Activity Schema

The Application Activity class captures application-level events and operations, including web application requests, API calls, database queries, email activities, file sharing operations, and other application-specific events. This schema is part of the OCSF (Open Cybersecurity Schema Framework) standard. For general OCSF guidance and field naming conventions, see "OCSF Schema Overview" on page 39.

For more information on the OCSF Version 1.2 standard, see OCSF v1.2 Documentation.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Field	Description	Value Type
action	Optional: The normalized caption of 'action_id' or the source-specific action.	string

Field	Description	Value Type
action_id	Required: The normalized action taken by a control or other policy-based system leading to an outcome or disposition.	integer
activity_id	Required: The normalized identifier of the activity that triggered the event.	integer
activity_name	Optional: The event activity name, as defined by the activity_id.	string
actor	Required: The actor that performed the activity on the target file.	object
actor.app_uid	Optional: The unique identifier of the client application or service that initiated the activity.	string
actor.idp	Optional: This object describes details about the Identity Provider used.	object
actor.idp.name	Recommended: The name of the identity provider.	string
actor.idp.uid	Recommended: The unique identifier of the identity provider.	string
actor.invoked_by	Optional: The name of the service that invoked the activity as described in the event.	string
actor.process	Recommended: The process that initiated the activity.	object
actor.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string

Field	Description	Value Type
actor.process.container	Recommended: The information describing an instance of a container.	object
actor.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861....`	object
actor.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.container.name	Recommended: The container name.	string
actor.process.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
actor.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string
actor.process.created_time	Recommended: The time when the process was created/started.	long
actor.process.file	Recommended: The process file object.	object
actor.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
actor.process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.file.name	Required: The name of the file.	string
actor.process.file.parent_folder	Optional: The parent folder in which the file resides.	string
actor.process.file.path	Recommended: The full path to the file. .	string
actor.process.file.size	Optional: The size of data, in bytes.	long
actor.process.file.type	Optional: The file type.	string
actor.process.file.type_id	Required: The file type ID.	integer
actor.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process	Recommended: The parent process of this process object.	object

Field	Description	Value Type
actor.process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.container	Recommended: The information describing an instance of a container.	object
actor.process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5...`.	object
actor.process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
actor.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0df....`	string
actor.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.file.name	Required: The name of the file.	string

Field	Description	Value Type
actor.process.parent_process.file.path	Recommended: The full path to the file.	string
actor.process.parent_process.file.type	Optional: The file type.	string
actor.process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
actor.process.parent_ process.parent_ process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2...` .	object
actor.process.parent_ process.parent_ process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_ process.parent_ process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_ process.parent_ process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_ process.parent_ process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_ process.parent_ process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_ process.parent_ process.container.name	Recommended: The container name.	string
actor.process.parent_ process.parent_ process.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
actor.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc...`.	string
actor.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
actor.process.parent_process.parent_process.file.name	Required: The name of the file.	string
actor.process.parent_process.parent_process.file.path	Recommended: The full path to the file.	string
actor.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier.	string

Field	Description	Value Type
actor.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_ process.session	Optional: The user session under which this process is running.	object
actor.process.parent_ process.session.created_time	Recommended: The time when the session was created.	long
actor.process.parent_ process.session.is_remote	Recommended: The indication of whether the session is remote.	boolea n
actor.process.parent_ process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.parent_ process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_ process.user	Recommended: The user under which this process is running.	object
actor.process.parent_ process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.parent_ process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_ process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_ process.user.uid	Recommended: The unique user identifier.	string

Field	Description	Value Type
actor.process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.session	Optional: The user session under which this process is running.	object
actor.process.session.created_time	Recommended: The time when the session was created.	long
actor.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.session.uuid	Optional: The universally unique identifier of the session.	string
actor.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.user	Recommended: The user under which this process is running.	object
actor.process.user.account	Optional: The user's account or the account associated with the user.	object
actor.process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string

Field	Description	Value Type
actor.process.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.user.email_addr	Optional: The user's primary email address.	string
actor.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.user.org	Optional: Organization and org unit related to the user.	object
actor.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string

Field	Description	Value Type
actor.process.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string
actor.process.user.type_id	Recommended: The account type identifier.	integer
actor.process.user.uid	Recommended: The unique user identifier.	string
actor.session	Optional: The user session from which the activity was initiated.	object
actor.session.created_time	Recommended: The time when the session was created.	long
actor.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.session.issuer	Recommended: The identifier of the session issuer.	string
actor.session.uid	Recommended: The unique identifier of the session.	string
actor.session.uuid	Optional: The universally unique identifier of the session.	string
actor.user	Recommended: The user that initiated the activity or the user context from which the activity was initiated.	object
actor.user.account	Optional: The user's account or the account associated with the user.	object
actor.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string

Field	Description	Value Type
actor.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.user.email_addr	Optional: The user's primary email address.	string
actor.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.user.org	Optional: Organization and org unit related to the user.	object
actor.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string

Field	Description	Value Type
actor.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string
actor.user.type_id	Recommended: The account type identifier.	integer
actor.user.uid	Recommended: The unique user identifier.	string
api	Optional: Describes details about a typical API (Application Programming Interface) call.	object
api.operation	Required: Verb/Operation associated with the request	string
api.request	Recommended: Details pertaining to the API request.	object
api.request.uid	Required: The unique request identifier.	string
api.response	Recommended: Details pertaining to the API response.	object
api.response.code	Recommended: The numeric response sent to a request.	integer
api.response.error	Recommended: Error Code	string
api.response.error_message	Recommended: Error Message	string
api.response.message	Recommended: The description of the event/finding, as defined by the source.	string
api.service	Optional: The information pertaining to the API service.	object

Field	Description	Value Type
api.service.name	Recommended: The name of the service.	string
api.service.uid	Recommended: The unique identifier of the service.	string
api.service.version	Recommended: The version of the service.	string
app	Required: The application that was affected by the lifecycle event. This also applies to self-updating application systems.	object
app.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
app.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
app.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
app.feature	Optional: The feature that reported the event.	object
app.feature.name	Recommended: The name of the feature.	string
app.feature.uid	Recommended: The unique identifier of the feature.	string
app.feature.version	Recommended: The version of the feature.	string

Field	Description	Value Type
app.name	Recommended: The name of the product.	string
app.uid	Recommended: The unique identifier of the product.	string
app.vendor_name	Required: The name of the vendor of the product.	string
app.version	Recommended: The version of the product, as defined by the event source. For example: `2013.1.3-beta`.	string
category_name	Optional: The event category name, as defined by category_uid value.	string
category_uid	Required: The category unique identifier of the event.	integer
class_name	Optional: The event class name, as defined by the class_uid.	string
class_uid	Required: The unique identifier of a class. A class describes the attributes available in an event.	integer
cloud	Required: Describes details about the Cloud environment where the event was originally created or logged.	object
cloud.account	Optional: The account object describes details about the account that was the source or target of the activity.	object
cloud.account.name	Recommended: The name of the account (For example, GCP Account Name).	string

Field	Description	Value Type
cloud.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
cloud.account.type_id	Recommended: The normalized account type identifier.	integer
cloud.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
cloud.org	Optional: Organization and org unit relevant to the event or object.	object
cloud.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
cloud.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
cloud.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
cloud.provider	Required: The unique name of the Cloud services provider, such as AWS, MS Azure, GCP,	string
cloud.region	Recommended: The name of the cloud region, as defined by the cloud provider.	string

Field	Description	Value Type
command_uid	Recommended: The command identifier that is associated with this scan event. This ID uniquely identifies the proactive scan command, For example, , if remotely initiated.	string
connection_info	Optional: The network connection information.	object
connection_info.direction	Optional: The direction of the initiated connection, traffic, or email, normalized to the caption of the direction_id value.	string
connection_info.direction_id	Required: The normalized identifier of the direction of the initiated connection, traffic, or email.	integer
connection_info.protocol_name	Optional: The TCP/IP protocol name in lowercase, as defined by the Internet Assigned Numbers Authority (IANA). .	string
connection_info.protocol_num	Recommended: The TCP/IP protocol number, as defined by the Internet Assigned Numbers Authority (IANA).	integer
connection_info.session	Optional: The authenticated user or service session.	object
connection_info.session.created_time	Recommended: The time when the session was created.	long
connection_info.session.is_remote	Recommended: The indication of whether the session is remote.	boolean

Field	Description	Value Type
connection_info.session.issuer	Recommended: The identifier of the session issuer.	string
connection_info.session.uid	Recommended: The unique identifier of the session.	string
connection_info.session.uuid	Optional: The universally unique identifier of the session.	string
connection_info.uid	Optional: The unique identifier of the connection.	string
count	Optional: The number of times that events in the same logical group occurred during the event Start Time to End Time period.	integer
database	Recommended: The database object is used for databases which are typically datastore services that contain an organized collection of structured and unstructured data or a types of data.	object
database.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
database.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
database.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer

Field	Description	Value Type
database.name	Recommended: The database name, ordinarily as assigned by a database administrator.	string
database.type	Recommended: The database type.	string
database.type_id	Required: The normalized identifier of the database type.	integer
database.uid	Recommended: The unique identifier of the database.	string
databucket	Recommended: The data bucket object is a basic container that holds data, typically organized through the use of data partitions.	object
databucket.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
databucket.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
databucket.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
databucket.name	Recommended: The databucket name.	string
databucket.type	Recommended: The databucket type.	string
databucket.type_id	Required: The normalized identifier of the databucket type.	integer

Field	Description	Value Type
databucket.uid	Recommended: The unique identifier of the databucket.	string
device	Recommended: An addressable device, computer system or host.	object
device.container	Recommended: The information describing an instance of a container.	object
device.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e....`	object
device.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
device.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
device.container.hash.value	Required: The digital fingerprint value.	string
device.container.image	Recommended: The container image used as a template to run the container.	object
device.container.image.name	Recommended: The image name. For example: `elixir`.	string
device.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string

Field	Description	Value Type
device.container.name	Recommended: The container name.	string
device.container.size	Recommended: The size of the container image.	long
device.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0...`.	string
device.domain	Optional: The network domain where the device resides. For example: `work.example.com`.	string
device.groups	Optional: The group names to which the device belongs. For example: `["Windows Laptops", "Engineering"]`.	array of objects
device.groups.desc	Optional: The group description.	string
device.groups.domain	Optional: The domain where the group is defined. For example: the LDAP or Active Directory domain.	string
device.groups.name	Recommended: The group name.	string
device.groups.type	Optional: The type of the group or account.	string
device.groups.uid	Recommended: The unique identifier of the group. For example, for Windows events this is the security identifier (SID) of the group.	string
device.hostname	Recommended: The device hostname.	string

Field	Description	Value Type
device.instance_uid	Recommended: The unique identifier of a VM instance.	string
device.interface_name	Recommended: The name of the network interface (For example, eth2).	string
device.interface_uid	Recommended: The unique identifier of the network interface.	string
device.ip	Recommended: The device IP address, in either IPv4 or IPv6 format.	string
device.is_compliant	Optional: The event occurred on a compliant device.	boolean
device.is_managed	Optional: The event occurred on a managed device.	boolean
device.name	Recommended: The alternate device name, ordinarily as assigned by an administrator. Name could be any other string that helps to identify the device, such as a phone number.	string
device.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
device.os	Optional: The endpoint operating system.	object
device.os.name	Required: The operating system name.	string
device.os.type	Optional: The type of the operating system.	string

Field	Description	Value Type
device.os.type_id	Required: The type identifier of the operating system.	integer
device.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
device.owner.account	Optional: The user's account or the account associated with the user.	object
device.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
device.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
device.owner.account.type_id	Recommended: The normalized account type identifier.	integer
device.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
device.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
device.owner.email_addr	Optional: The user's primary email address.	string
device.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
device.owner.name	Recommended: The username. For example, `janedoe1`.	string

Field	Description	Value Type
device.owner.org	Optional: Organization and org unit related to the user.	object
device.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
device.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
device.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
device.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
device.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
device.owner.type_id	Recommended: The account type identifier.	integer
device.owner.uid	Recommended: The unique user identifier.	string
device.region	Recommended: The region where the virtual machine is located. For example, an AWS Region.	string
device.type	Optional: The device type.	string
device.type_id	Required: The device type ID.	integer

Field	Description	Value Type
device.uid	Recommended: The unique identifier of the device. For example the Windows TargetSID or AWS EC2 ARN.	string
dst_endpoint	Recommended: The endpoint that received the activity on the target file.	object
dst_endpoint.container	Recommended: The information describing an instance of a container.	object
dst_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5...`.	object
dst_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string

Field	Description	Value Type
dst_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
dst_endpoint.container.name	Recommended: The container name.	string
dst_endpoint.container.size	Recommended: The size of the container image.	long
dst_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string
dst_endpoint.domain	Optional: The name of the domain.	string
dst_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
dst_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
dst_endpoint.interface_name	Recommended: The name of the network interface.	string
dst_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.location	Optional: The geographical location of the endpoint.	object
dst_endpoint.location.city	Recommended: The name of the city.	string
dst_endpoint.location.continent	Recommended: The name of the continent.	string

Field	Description	Value Type
dst_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code. The two letter country code should be capitalized.	string
dst_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD).	float
dst_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD).	float
dst_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision of the country.	string
dst_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
dst_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
dst_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
dst_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object

Field	Description	Value Type
dst_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
dst_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
dst_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
dst_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
dst_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
dst_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
dst_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
dst_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string

Field	Description	Value Type
dst_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
dst_endpoint.owner.type_id	Recommended: The account type identifier.	integer
dst_endpoint.owner.uid	Recommended: The unique user identifier.	string
dst_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
dst_endpoint.proxy_ endpoint.container	Recommended: The information describing an instance of a container.	object
dst_endpoint.proxy_ endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2...` .	object

Field	Description	Value Type
dst_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
dst_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
dst_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
dst_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
dst_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc8...`.	string
dst_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
dst_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string

Field	Description	Value Type
dst_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
dst_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
dst_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
dst_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
dst_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
dst_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
dst_endpoint.proxy_ endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
dst_endpoint.proxy_ endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.proxy_ endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.proxy_ endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
dst_endpoint.proxy_ endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
dst_endpoint.proxy_ endpoint.owner.org	Optional: Organization and org unit related to the user.	object
dst_endpoint.proxy_ endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
dst_endpoint.proxy_ endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
dst_endpoint.proxy_ endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.proxy_ endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.proxy_ endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string

Field	Description	Value Type
dst_endpoint.proxy_ endpoint.owner.type_id	Recommended: The account type identifier.	integer
dst_endpoint.proxy_ endpoint.owner.uid	Recommended: The unique user identifier.	string
dst_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint.svc_ name	Recommended: The service name in service-to-service connections.	string
dst_endpoint.proxy_ endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
dst_endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.zone	Optional: The network zone or LAN segment.	string
duration	Recommended: The duration of the scan	integer
end_time	Optional: The end time of a time period, or the time of the most recent event included in the aggregate event.	long

Field	Description	Value Type
event_seq_id	Required: The sequence identifier for the event.	long
event_size	Required: The size of the event in bytes.	long
event_time	Required: The normalized event occurrence time or the finding creation time.	long
file	Required: The file that is the target of the activity.	object
file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
file.data_ classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
file.hashes	Recommended: An array of hash attributes.	array of objects
file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
file.hashes.value	Required: The digital fingerprint value.	string
file.name	Required: The name of the file.	string
file.parent_folder	Optional: The parent folder in which the file resides.	string
file.path	Recommended: The full path to the file. .	string
file.size	Optional: The size of data, in bytes.	long
file.type	Optional: The file type.	string
file.type_id	Required: The file type ID.	integer
http_request	Recommended: Details about the underlying http request.	object
http_request.http_headers	Recommended: Additional HTTP headers of an HTTP request or response.	array of objects
http_request.http_headers.name	Required: The name of the header	string
http_request.http_headers.value	Required: The value of the header	string
http_request.http_method	Optional: The HTTP request method indicates the desired action to be performed for a given resource.	string
http_request.length	Optional: The HTTP request length, in number of bytes.	integer
http_request.referrer	Optional: The request header that identifies the address of the previous web page, which is linked to the current web page or resource being requested.	string

Field	Description	Value Type
http_request.uid	Optional: The unique identifier of the http request.	string
http_request.url	Recommended: The URL object that pertains to the request.	object
http_request.url.category_ids	Recommended: The Website categorization identifiers.	array of integer s
http_request.url.hostname	Recommended: The URL host as extracted from the URL. .	string
http_request.url.path	Recommended: The URL path as extracted from the URL.	string
http_request.url.port	Recommended: The URL port. For example: `80`.	integer
http_request.url.query_string	Recommended: The query portion of the URL.	string
http_request.url.scheme	Recommended: The scheme portion of the URL.	string
http_request.url.url_string	Recommended: The URL string. See RFC 1738.	string
http_request.user_agent	Recommended: The request header that identifies the operating system and web browser.	string
http_request.version	Recommended: The Hypertext Transfer Protocol (HTTP) version.	string
http_response	Optional: Details about the HTTP response, if available.	object

Field	Description	Value Type
http_response.code	Required: The Hypertext Transfer Protocol (HTTP) status code returned from the web server to the client.	integer
http_response.content_type	Optional: The request header that identifies the original media type of the resource (prior to any content encoding applied for sending).	string
http_response.http_headers	Recommended: Additional HTTP headers of an HTTP request or response.	array of objects
http_response.http_headers.name	Required: The name of the header	string
http_response.http_headers.value	Required: The value of the header	string
indexed_time	Required: The time when the event was indexed.	long
message	Recommended: The description of the event/finding, as defined by the source.	string
metadata	Required: The metadata associated with the event or a finding.	object
metadata.correlation_uid	Optional: The unique identifier used to correlate events.	string
metadata.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
metadata.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.event_code	Optional: The Event ID or Code that the product uses to describe the event.	string
metadata.log_level	Optional: The audit level at which an event was generated.	string
metadata.log_name	Recommended: The event log name. For example, syslog file name or Windows logging subsystem: Security.	string
metadata.log_provider	Recommended: The logging provider or logging service that logged the event. For example, Microsoft-Windows-Security-Auditing.	string
metadata.original_time	Recommended: The original event time as reported by the event source.	string
metadata.product	Required: The product that reported the event.	object
metadata.product.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
metadata.product.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.product.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.product.feature	Optional: The feature that reported the event.	object
metadata.product.feature.name	Recommended: The name of the feature.	string
metadata.product.feature.uid	Recommended: The unique identifier of the feature.	string
metadata.product.feature.version	Recommended: The version of the feature.	string
metadata.product.name	Recommended: The name of the product.	string
metadata.product.uid	Recommended: The unique identifier of the product.	string
metadata.product.vendor_name	Required: The name of the vendor of the product.	string
metadata.product.version	Recommended: The version of the product, as defined by the event source. For example: `2013.1.3-beta`.	string
metadata.profiles	Optional: The list of profiles used to create the event.	array of strings

Field	Description	Value Type
metadata.sequence	Optional: Sequence number of the event. The sequence number is a value available in some events, to make the exact ordering of events unambiguous, regardless of the event time precision.	integer
metadata.tenant_uid	Recommended: The unique tenant identifier.	string
metadata.uid	Optional: The logging system-assigned unique identifier of an event instance.	string
metadata.version	Required: The version of the OCSF schema, using Semantic Versioning Specification.	string
num_detections	Recommended: The number of detections.	integer
num_files	Recommended: The number of files scanned.	integer
num_folders	Recommended: The number of folders scanned.	integer
num_network_items	Recommended: The number of network items scanned.	integer
num_processes	Recommended: The number of processes scanned.	integer
num_registry_items	Recommended: The number of registry items scanned.	integer
num_resolutions	Recommended: The number of items that were resolved.	integer
num_skipped_items	Recommended: The number of skipped items.	integer

Field	Description	Value Type
num_trusted_items	Recommended: The number of trusted items.	integer
observables	Recommended: The observables associated with the event or a finding.	array of objects
observables.name	Required: The full name of the observable attribute. The `name` is a pointer/reference to an attribute within the event data. For example: `file.name`.	string
observables.type	Optional: The observable value type name.	string
observables.type_id	Required: The observable value type identifier.	integer
observables.value	Optional: The value associated with the observable attribute. The meaning of the value depends on the observable type.	string
original_event_time	Required: The original event time from the source system.	long
policy	Recommended: The policy associated with this Scan event; required if the scan was initiated by a policy.	object
policy.name	Recommended: The policy name. For example: `IAM Policy`.	string
policy.uid	Recommended: A unique identifier of the policy instance.	string
policy.version	Recommended: The policy version number.	string

Field	Description	Value Type
proxy_connection_info	Recommended: The connection information from the proxy server to the remote server.	object
proxy_connection_info.direction	Optional: The direction of the initiated connection, traffic, or email, normalized to the caption of the direction_id value.	string
proxy_connection_info.direction_id	Required: The normalized identifier of the direction of the initiated connection, traffic, or email.	integer
proxy_connection_info.protocol_name	Optional: The TCP/IP protocol name in lowercase, as defined by the Internet Assigned Numbers Authority (IANA).	string
proxy_connection_info.protocol_num	Recommended: The TCP/IP protocol number, as defined by the Internet Assigned Numbers Authority (IANA).	integer
proxy_connection_info.session	Optional: The authenticated user or service session.	object
proxy_connection_info.session.created_time	Recommended: The time when the session was created.	long
proxy_connection_info.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
proxy_connection_info.session.issuer	Recommended: The identifier of the session issuer.	string
proxy_connection_info.session.uid	Recommended: The unique identifier of the session.	string
proxy_connection_info.session.uuid	Optional: The universally unique identifier of the session.	string

Field	Description	Value Type
proxy_connection_info.uid	Optional: The unique identifier of the connection.	string
proxy_tls	Recommended: The TLS protocol negotiated between the proxy server and the remote server.	object
proxy_tls.certificate	Recommended: The certificate object containing information about the digital certificate.	object
proxy_tls.certificate.created_time	Recommended: The time when the certificate was created.	long
proxy_tls.certificate.expiration_time	Recommended: The expiration time of the certificate.	long
proxy_tls.certificate.fingerprints	Required: The fingerprint list of the certificate.	array of objects
proxy_tls.certificate.fingerprints.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
proxy_tls.certificate.fingerprints.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
proxy_tls.certificate.fingerprints.value	Required: The digital fingerprint value.	string
proxy_tls.certificate.issuer	Required: The certificate issuer distinguished name.	string
proxy_tls.certificate.serial_number	Required: The serial number of the certificate used to create the digital signature.	string

Field	Description	Value Type
proxy_tls.certificate.subject	Recommended: The certificate subject distinguished name.	string
proxy_tls.certificate.version	Recommended: The certificate version.	string
proxy_tls.certificate_chain	Recommended: The Chain of Certificate Serial Numbers field provides a chain of Certificate Issuer Serial Numbers leading to the Root Certificate Issuer.	array of strings
proxy_tls.cipher	Recommended: The negotiated cipher suite.	string
proxy_tls.client_ciphers	Recommended: The client cipher suites that were exchanged during the TLS handshake negotiation.	array of strings
proxy_tls.ja3_hash	Recommended: The MD5 hash of a JA3 string.	object
proxy_tls.ja3_hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
proxy_tls.ja3_hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
proxy_tls.ja3_hash.value	Required: The digital fingerprint value.	string
proxy_tls.ja3s_hash	Recommended: The MD5 hash of a JA3S string.	object

Field	Description	Value Type
proxy_tls.ja3s_hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
proxy_tls.ja3s_hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
proxy_tls.ja3s_hash.value	Required: The digital fingerprint value.	string
proxy_tls.sni	Recommended: The Server Name Indication (SNI) extension sent by the client.	string
proxy_tls.version	Required: The TLS protocol version.	string
proxy_traffic	Recommended: The network traffic refers to the amount of data moving across a network, from proxy to remote server at a given point of time.	object
proxy_traffic.bytes	Recommended: The total number of bytes (in and out).	long
proxy_traffic.bytes_in	Optional: The number of bytes sent from the destination to the source.	long
proxy_traffic.bytes_out	Optional: The number of bytes sent from the source to the destination.	long
proxy_traffic.chunks	Optional: The total number of chunks (in and out).	long

Field	Description	Value Type
proxy_traffic.chunks_out	Optional: The number of chunks sent from the source to the destination.	long
proxy_traffic.packets	Recommended: The total number of packets (in and out).	long
proxy_traffic.packets_in	Optional: The number of packets sent from the destination to the source.	long
proxy_traffic.packets_out	Optional: The number of packets sent from the source to the destination.	long
query_info	Recommended: The query info object holds information related to data access within a datastore.	object
query_info.name	Recommended: The query name for a saved or scheduled query.	string
query_info.query_string	Required: A string representing the query code being run.	string
query_info.uid	Recommended: The unique identifier of the query.	string
raw_data	Optional: The raw event/finding data as received from the source.	string
raw_data_size	Optional: The size of the raw data which was transformed into an OCSF event, in bytes.	long
resources	Recommended: Details about resources that were affected by the activity/event.	array of objects

Field	Description	Value Type
resources.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
resources.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
resources.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
resources.owner	Recommended: The identity of the service or user account that owns the resource.	object
resources.owner.account	Optional: The user's account or the account associated with the user.	object
resources.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
resources.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
resources.owner.account.type_id	Recommended: The normalized account type identifier.	integer
resources.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
resources.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string

Field	Description	Value Type
resources.owner.email_addr	Optional: The user's primary email address.	string
resources.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
resources.owner.name	Recommended: The username. For example, `janedoe1`.	string
resources.owner.org	Optional: Organization and org unit related to the user.	object
resources.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
resources.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
resources.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
resources.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
resources.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
resources.owner.type_id	Recommended: The account type identifier.	integer
resources.owner.uid	Recommended: The unique user identifier.	string

Field	Description	Value Type
scan	Required: The Scan object describes characteristics of the scan job.	object
scan.name	Recommended: The administrator-supplied or application-generated name of the scan.	string
scan.type_id	Required: The type id of the scan.	integer
scan.uid	Recommended: The application-defined unique identifier assigned to an instance of a scan.	string
schedule_uid	Recommended: The unique identifier of the schedule associated with a scan job.	string
severity	Optional: The event severity, normalized to the caption of the severity_id value.	string
severity_id	Required: The normalized identifier of the event/finding severity.	integer
src_endpoint	Required: The endpoint that performed the activity on the target file.	object
src_endpoint.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
src_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b44...`.	object
src_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
src_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
src_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
src_endpoint.container.name	Recommended: The container name.	string
src_endpoint.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
src_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff3...`.	string
src_endpoint.domain	Optional: The name of the domain.	string
src_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
src_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
src_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.location	Optional: The geographical location of the endpoint.	object
src_endpoint.location.city	Recommended: The name of the city.	string
src_endpoint.location.continent	Recommended: The name of the continent.	string
src_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string

Field	Description	Value Type
src_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD).	float
src_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD).	float
src_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (For example, province or state) of the country.	string
src_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
src_endpoint.name	Recommended: The short name of the endpoint.	string
src_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
src_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_ endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string

Field	Description	Value Type
src_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
src_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
src_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.owner.email_addr	Optional: The user's primary email address.	string
src_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
src_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
src_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string

Field	Description	Value Type
src_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User.	string
src_endpoint.owner.type_id	Recommended: The account type identifier.	integer
src_endpoint.owner.uid	Recommended: The unique user identifier.	string
src_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint. For example, .	object
src_endpoint.proxy_ endpoint.container	Recommended: The information describing an instance of a container.	object
src_endpoint.proxy_ endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b....`	object
src_endpoint.proxy_ endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string

Field	Description	Value Type
src_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
src_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
src_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
src_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
src_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff3...`.	string
src_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
src_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string

Field	Description	Value Type
src_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
src_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
src_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
src_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
src_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string

Field	Description	Value Type
src_endpoint.proxy_ endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.proxy_ endpoint.owner.email_addr	Optional: The user's primary email address.	string
src_endpoint.proxy_ endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
src_endpoint.proxy_ endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
src_endpoint.proxy_ endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.proxy_ endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.proxy_ endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.proxy_ endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
src_endpoint.proxy_ endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.proxy_ endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User.	string
src_endpoint.proxy_ endpoint.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
src_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier.	string
src_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.zone	Optional: The network zone or LAN segment.	string
start_time	Recommended: The start time of the scan job.	long
status	Recommended: The event status, normalized to the caption of the status_id value.	string
status_code	Recommended: The event status code, as reported by the event source.	string

Field	Description	Value Type
status_detail	Recommended: The status details contains additional information about the event/finding outcome.	string
status_id	Recommended: The normalized identifier of the event status.	integer
table	Recommended: The table object represents a table within a structured relational database or datastore, which contains columns and rows of data that are able to be create, updated, deleted and queried.	object
table.name	Recommended: The table name, ordinarily as assigned by a database administrator.	string
table.uid	Recommended: The unique identifier of the table.	string
timezone_offset	Recommended: The number of minutes that the reported event `time` is ahead or behind UTC, in the range -1,080 to +1,080.	integer
total	Recommended: The total number of items that were scanned; zero if no items were scanned.	integer
type_id	Recommended: The normalized datastore resource type identifier.	integer
type_name	Optional: The event type name, as defined by the type_uid.	string
type_uid	Required: The event/finding type ID. It identifies the event's semantics and structure.	long

Field	Description	Value Type
unmapped	Optional: The attributes that are not mapped to the event schema.	object
volume	Required: The allocated event volume for the event in bytes.	long
web_resources	Required: Describes details about web resources that were affected by an activity/event.	array of objects
web_resources.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
web_resources.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
web_resources.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
web_resources.url_string	Recommended: The URL pointing towards the source of the web resource.	string
web_resources_result	Recommended: The results of the activity on web resources.	array of objects
web_resources_result.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
web_resources_result.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
web_resources_result.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
web_resources_result.url_string	Recommended: The URL pointing towards the source of the web resource.	string

OCSF Discovery Schema

The Discovery class captures device and service discovery events, including asset inventory data, network scanning results, service enumeration, device profiling, and configuration discovery from various sources. This schema is part of the OCSF (Open Cybersecurity Schema Framework) standard. For general OCSF guidance and field naming conventions, see "OCSF Schema Overview" on page 39.

For more information on the OCSF Version 1.2 standard, see OCSF v1.2 Documentation.

Field	Description	Value Type
action	Optional: The normalized caption of 'action_id' or the source-specific action.	string
action_id	Required: The normalized action taken by a control or other policy-based system leading to an outcome or disposition.	integer
activity_id	Required: The normalized identifier of the activity that triggered the event.	integer
activity_name	Optional: The event activity name, as defined by the activity_id.	string

Field	Description	Value Type
actor	Optional: The actor object describes details about the user/role/process that was the source of the activity.	object
actor.app_uid	Optional: The unique identifier of the client application or service that initiated the activity.	string
actor.idp	Optional: This object describes details about the Identity Provider used.	object
actor.idp.name	Recommended: The name of the identity provider.	string
actor.idp.uid	Recommended: The unique identifier of the identity provider.	string
actor.invoked_by	Optional: The name of the service that invoked the activity as described in the event.	string
actor.process	Recommended: The process that initiated the activity.	object
actor.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.container	Recommended: The information describing an instance of a container.	object
actor.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e...`	object

Field	Description	Value Type
actor.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.container.name	Recommended: The container name.	string
actor.process.container.size	Recommended: The size of the container image.	long
actor.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff35...`.	string
actor.process.created_time	Recommended: The time when the process was created/started.	long
actor.process.file	Recommended: The process file object.	object

Field	Description	Value Type
actor.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.file.name	Required: The name of the file. For example: `svchost.exe`	string
actor.process.file.parent_folder	Optional: The parent folder in which the file resides.	string
actor.process.file.path	Recommended: The full path to the file.	string
actor.process.file.size	Optional: The size of data, in bytes.	long

Field	Description	Value Type
actor.process.file.type	Optional: The file type.	string
actor.process.file.type_id	Required: The file type ID.	integer
actor.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_ process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_ process.container	Recommended: The information describing an instance of a container.	object
actor.process.parent_ process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b4....`	object

Field	Description	Value Type
actor.process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0df....`	string
actor.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
actor.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string
actor.process.parent_process.file.path	Recommended: The full path to the file. .	string
actor.process.parent_process.file.type	Optional: The file type.	string
actor.process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer

Field	Description	Value Type
actor.process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object
actor.process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b4....`	object
actor.process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object

Field	Description	Value Type
actor.process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff...`.	string
actor.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
actor.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string
actor.process.parent_process.parent_process.file.path	Recommended: The full path to the file. .	string
actor.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer

Field	Description	Value Type
actor.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier.	string
actor.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.session	Optional: The user session under which this process is running.	object
actor.process.parent_process.session.created_time	Recommended: The time when the session was created.	long
actor.process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string

Field	Description	Value Type
actor.process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.user.uid	Recommended: The unique user identifier.	string
actor.process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.session	Optional: The user session under which this process is running.	object
actor.process.session.created_time	Recommended: The time when the session was created.	long
actor.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.session.uuid	Optional: The universally unique identifier of the session.	string
actor.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string

Field	Description	Value Type
actor.process.user	Recommended: The user under which this process is running.	object
actor.process.user.account	Optional: The user's account or the account associated with the user.	object
actor.process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.process.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.user.email_addr	Optional: The user's primary email address.	string
actor.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.user.org	Optional: Organization and org unit related to the user.	object
actor.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string

Field	Description	Value Type
actor.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.process.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
actor.process.user.type_id	Recommended: The account type identifier.	integer
actor.process.user.uid	Recommended: The unique user identifier.	string
actor.session	Optional: The user session from which the activity was initiated.	object
actor.session.created_time	Recommended: The time when the session was created.	long
actor.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.session.issuer	Recommended: The identifier of the session issuer.	string
actor.session.uid	Recommended: The unique identifier of the session.	string
actor.session.uuid	Optional: The universally unique identifier of the session.	string

Field	Description	Value Type
actor.user	Recommended: The user that initiated the activity or the user context from which the activity was initiated.	object
actor.user.account	Optional: The user's account or the account associated with the user.	object
actor.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.user.email_addr	Optional: The user's primary email address.	string
actor.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.user.org	Optional: Organization and org unit related to the user.	object

Field	Description	Value Type
actor.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
actor.user.type_id	Recommended: The account type identifier.	integer
actor.user.uid	Recommended: The unique user identifier.	string
api	Optional: Describes details about a typical API (Application Programming Interface) call.	object
api.operation	Required: Verb/Operation associated with the request	string
api.request	Recommended: Details pertaining to the API request.	object
api.request.uid	Required: The unique request identifier.	string

Field	Description	Value Type
api.response	Recommended: Details pertaining to the API response.	object
api.response.code	Recommended: The numeric response sent to a request.	integer
api.response.error	Recommended: Error Code	string
api.response.error_message	Recommended: Error Message	string
api.response.message	Recommended: The description of the event/finding, as defined by the source.	string
api.service	Optional: The information pertaining to the API service.	object
api.service.name	Recommended: The name of the service.	string
api.service.uid	Recommended: The unique identifier of the service.	string
api.service.version	Recommended: The version of the service.	string
category_name	Optional: The event category name, as defined by category_uid value.	string
category_uid	Required: The category unique identifier of the event.	integer
cis_benchmark_result	Recommended: The CIS benchmark result.	object
cis_benchmark_result.name	Required: The CIS benchmark name.	string
class_name	Optional: The event class name, as defined by the class_uid.	string

Field	Description	Value Type
class_uid	Required: The unique identifier of a class.	integer
cloud	Required: Describes details about the Cloud environment where the event was originally created or logged.	object
cloud.account	Optional: The account object describes details about the account that was the source or target of the activity.	object
cloud.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
cloud.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
cloud.account.type_id	Recommended: The normalized account type identifier.	integer
cloud.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
cloud.org	Optional: Organization and org unit relevant to the event or object.	object
cloud.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
cloud.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string

Field	Description	Value Type
cloud.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
cloud.provider	Required: The unique name of the Cloud services provider, such as AWS, MS Azure, GCP, etc.	string
cloud.region	Recommended: The name of the cloud region, as defined by the cloud provider.	string
connection_info	Required: The network connection information.	object
connection_info.direction	Optional: The direction of the initiated connection, traffic, or email, normalized to the caption of the direction_id value.	string
connection_info.direction_id	Required: The normalized identifier of the direction of the initiated connection, traffic, or email.	integer
connection_info.protocol_name	Optional: The TCP/IP protocol name in lowercase, as defined by the Internet Assigned Numbers Authority (IANA). For example: `tcp` or `udp`.	string
connection_info.protocol_num	Recommended: The TCP/IP protocol number, as defined by the Internet Assigned Numbers Authority (IANA). For example: `6` for TCP and `17` for UDP.	integer
connection_info.session	Optional: The authenticated user or service session.	object

Field	Description	Value Type
connection_info.session.created_time	Recommended: The time when the session was created.	long
connection_info.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
connection_info.session.issuer	Recommended: The identifier of the session issuer.	string
connection_info.session.uid	Recommended: The unique identifier of the session.	string
connection_info.session.uuid	Optional: The universally unique identifier of the session.	string
connection_info.uid	Optional: The unique identifier of the connection.	string
count	Optional: The number of times that events in the same logical group occurred during the event Start Time to End Time period.	integer
device	Recommended: An addressable device, computer system or host.	object
device.container	Recommended: The information describing an instance of a container.	object
device.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5....`	object
device.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string

Field	Description	Value Type
device.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
device.container.hash.value	Required: The digital fingerprint value.	string
device.container.image	Recommended: The container image used as a template to run the container.	object
device.container.image.name	Recommended: The image name. For example: `elixir`.	string
device.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
device.container.name	Recommended: The container name.	string
device.container.size	Recommended: The size of the container image.	long
device.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string
device.domain	Optional: The network domain where the device resides. For example: `work.example.com`.	string
device.groups	Optional: The group names to which the device belongs. For example: `["Windows Laptops", "Engineering"]`.	array of objects
device.groups.desc	Optional: The group description.	string

Field	Description	Value Type
device.groups.domain	Optional: The domain where the group is defined. For example: the LDAP or Active Directory domain.	string
device.groups.name	Recommended: The group name.	string
device.groups.type	Optional: The type of the group or account.	string
device.groups.uid	Recommended: The unique identifier of the group.	string
device.hostname	Recommended: The device hostname.	string
device.instance_uid	Recommended: The unique identifier of a VM instance.	string
device.interface_name	Recommended: The name of the network interface (For example, eth2).	string
device.interface_uid	Recommended: The unique identifier of the network interface.	string
device.ip	Recommended: The device IP address, in either IPv4 or IPv6 format.	string
device.is_compliant	Optional: The event occurred on a compliant device.	boolean
device.is_managed	Optional: The event occurred on a managed device.	boolean
device.name	Recommended: The alternate device name, ordinarily as assigned by an administrator.	string

Field	Description	Value Type
device.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
device.os	Optional: The endpoint operating system.	object
device.os.name	Required: The operating system name.	string
device.os.type	Optional: The type of the operating system.	string
device.os.type_id	Required: The type identifier of the operating system.	integer
device.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
device.owner.account	Optional: The user's account or the account associated with the user.	object
device.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
device.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
device.owner.account.type_id	Recommended: The normalized account type identifier.	integer
device.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string

Field	Description	Value Type
device.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
device.owner.email_addr	Optional: The user's primary email address.	string
device.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
device.owner.name	Recommended: The username. For example, `janedoe1`.	string
device.owner.org	Optional: Organization and org unit related to the user.	object
device.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
device.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
device.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
device.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
device.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
device.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
device.owner.uid	Recommended: The unique user identifier.	string
device.region	Recommended: The region where the virtual machine is located. For example, an AWS Region.	string
device.type	Optional: The device type. For example: `unknown`, `server`, `desktop`, `laptop`, `tablet`, `mobile`, `virtual`, `browser`, or `other`.	string
device.type_id	Required: The device type ID.	integer
device.uid	Recommended: The unique identifier of the device. For example the Windows TargetSID or AWS EC2 ARN.	string
duration	Optional: The event duration or aggregate time, the amount of time the event covers from `start_time` to `end_time` in milliseconds.	integer
end_time	Optional: The end time of a time period, or the time of the most recent event included in the aggregate event.	long
event_seq_id	Required: The sequence identifier for the event.	long
event_size	Required: The size of the event in bytes.	long
event_time	Required: The normalized event occurrence time or the finding creation time.	long

Field	Description	Value Type
file	Required: The file that is the target of the query.	object
file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
file.hashes	Recommended: An array of hash attributes.	array of objects
file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
file.hashes.value	Required: The digital fingerprint value.	string
file.name	Required: The name of the file. For example: `svchost.exe`	string
file.parent_folder	Optional: The parent folder in which the file resides.	string
file.path	Recommended: The full path to the file. .	string

Field	Description	Value Type
file.size	Optional: The size of data, in bytes.	long
file.type	Optional: The file type.	string
file.type_id	Required: The file type ID.	integer
folder	Required: The folder that is the target of the query.	object
folder.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
folder.data_ classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
folder.data_ classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
folder.hashes	Recommended: An array of hash attributes.	array of objects
folder.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
folder.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
folder.hashes.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
folder.name	Required: The name of the file. For example: `svchost.exe`	string
folder.parent_folder	Optional: The parent folder in which the file resides.	string
folder.path	Recommended: The full path to the file. .	string
folder.size	Optional: The size of data, in bytes.	long
folder.type	Optional: The file type.	string
folder.type_id	Required: The file type ID.	integer
group	Required: The administrative group.	object
group.desc	Optional: The group description.	string
group.domain	Optional: The domain where the group is defined. For example: the LDAP or Active Directory domain.	string
group.name	Recommended: The group name.	string
group.type	Optional: The type of the group or account.	string
group.uid	Recommended: The unique identifier of the group.	string
indexed_time	Required: The time when the event was indexed.	long
job	Required: The job object that pertains to the event.	object
job.cmd_line	Recommended: The job command line.	string

Field	Description	Value Type
job.created_time	Recommended: The time when the job was created.	long
job.desc	Recommended: The description of the job.	string
job.file	Required: The file that pertains to the job.	object
job.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
job.file.data_ classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
job.file.data_ classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
job.file.hashes	Recommended: An array of hash attributes.	array of objects
job.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
job.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
job.file.hashes.value	Required: The digital fingerprint value.	string
job.file.name	Required: The name of the file. For example: `svchost.exe`	string

Field	Description	Value Type
job.file.parent_folder	Optional: The parent folder in which the file resides.	string
job.file.path	Recommended: The full path to the file. .	string
job.file.size	Optional: The size of data, in bytes.	long
job.file.type	Optional: The file type.	string
job.file.type_id	Required: The file type ID.	integer
job.last_run_time	Recommended: The time when the job was last run.	long
job.name	Required: The name of the job.	string
job.run_state_id	Recommended: The run state ID of the job.	integer
kb_article_list	Recommended: A list of KB articles or patches related to an endpoint.	array of objects
kb_article_list.os	Recommended: The operating system the kb article applies.	object
kb_article_list.os.name	Required: The operating system name.	string
kb_article_list.os.type	Optional: The type of the operating system.	string
kb_article_list.os.type_id	Required: The type identifier of the operating system.	integer
kb_article_list.severity	Recommended: The severity of the kb article.	string
kb_article_list.title	Recommended: The title of the kb article.	string

Field	Description	Value Type
kb_article_list.uid	Required: The unique identifier for the kb article.	string
kernel	Required: The kernel object that pertains to the event.	object
kernel.name	Required: The name of the kernel resource.	string
kernel.type_id	Required: The type of the kernel resource.	integer
message	Recommended: The description of the event/finding, as defined by the source.	string
metadata	Required: The metadata associated with the event or a finding.	object
metadata.correlation_uid	Optional: The unique identifier used to correlate events.	string
metadata.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.event_code	Optional: The Event ID or Code that the product uses to describe the event.	string

Field	Description	Value Type
metadata.log_level	Optional: The audit level at which an event was generated.	string
metadata.log_name	Recommended: The event log name. For example, syslog file name or Windows logging subsystem: Security.	string
metadata.log_provider	Recommended: The logging provider or logging service that logged the event. For example, Microsoft-Windows-Security-Auditing.	string
metadata.original_time	Recommended: The original event time as reported by the event source.	string
metadata.product	Required: The product that reported the event.	object
metadata.product.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.product.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.product.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.product.feature	Optional: The feature that reported the event.	object
metadata.product.feature.name	Recommended: The name of the feature.	string

Field	Description	Value Type
metadata.product.feature.uid	Recommended: The unique identifier of the feature.	string
metadata.product.feature.version	Recommended: The version of the feature.	string
metadata.product.name	Recommended: The name of the product.	string
metadata.product.uid	Recommended: The unique identifier of the product.	string
metadata.product.vendor_name	Required: The name of the vendor of the product.	string
metadata.product.version	Recommended: The version of the product, as defined by the event source. For example: `2013.1.3-beta`.	string
metadata.profiles	Optional: The list of profiles used to create the event.	array of strings
metadata.sequence	Optional: Sequence number of the event.	integer
metadata.tenant_uid	Recommended: The unique tenant identifier.	string
metadata.uid	Optional: The logging system-assigned unique identifier of an event instance.	string
metadata.version	Required: The version of the OCSF schema, using Semantic Versioning Specification.	string
module	Required: The module that pertains to the event.	object

Field	Description	Value Type
module.base_address	Recommended: The memory address where the module was loaded.	string
module.file	Recommended: The module file object.	object
module.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
module.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
module.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
module.file.hashes	Recommended: An array of hash attributes.	array of objects
module.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
module.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
module.file.hashes.value	Required: The digital fingerprint value.	string
module.file.name	Required: The name of the file. For example: `svchost.exe`	string

Field	Description	Value Type
module.file.parent_folder	Optional: The parent folder in which the file resides.	string
module.file.path	Recommended: The full path to the file. .	string
module.file.size	Optional: The size of data, in bytes.	long
module.file.type	Optional: The file type.	string
module.file.type_id	Required: The file type ID.	integer
module.load_type_id	Required: The normalized identifier of the load type.	integer
module.start_address	Recommended: The start address of the execution.	string
module.type	Recommended: The module type.	string
network_interfaces	Required: The network interfaces that are associated with the device, one for each unique MAC address/IP address/hostname/name combination.The first element of the array is the network information that pertains to the event.	array of objects
network_interfaces.hostname	Recommended: The hostname associated with the network interface.	string
network_interfaces.ip	Recommended: The IP address associated with the network interface.	string
network_interfaces.mac	Recommended: The MAC address of the network interface.	string

Field	Description	Value Type
network_interfaces.name	Recommended: The name of the network interface.	string
network_interfaces.type_id	Required: The network interface type identifier.	integer
observables	Recommended: The observables associated with the event or a finding.	array of objects
observables.name	Required: The full name of the observable attribute. The `name` is a pointer/reference to an attribute within the event data. For example: `file.name`.	string
observables.type	Optional: The observable value type name.	string
observables.type_id	Required: The observable value type identifier.	integer
observables.value	Optional: The value associated with the observable attribute. The meaning of the value depends on the observable type.	string
original_event_time	Required: The original event time from the source system.	long
peripheral_device	Required: The peripheral device that triggered the event.	object
peripheral_device.class	Required: The class of the peripheral device.	string
peripheral_device.model	Recommended: The peripheral device model.	string
peripheral_device.name	Required: The name of the peripheral device.	string

Field	Description	Value Type
peripheral_device.serial_number	Recommended: The peripheral device serial number.	string
peripheral_device.uid	Recommended: The unique identifier of the peripheral device.	string
peripheral_device.vendor_name	Recommended: The peripheral device vendor.	string
process	Required: The process object.	object
process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
process.container	Recommended: The information describing an instance of a container.	object
process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2....`	object
process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.container.hash.value	Required: The digital fingerprint value.	string
process.container.image	Recommended: The container image used as a template to run the container.	object

Field	Description	Value Type
process.container.image.name	Recommended: The image name. For example: `elixir`.	string
process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
process.container.name	Recommended: The container name.	string
process.container.size	Recommended: The size of the container image.	long
process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82...`.	string
process.created_time	Recommended: The time when the process was created/started.	long
process.file	Recommended: The process file object.	object
process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
process.file.hashes	Recommended: An array of hash attributes.	array of objects

Field	Description	Value Type
process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.file.hashes.value	Required: The digital fingerprint value.	string
process.file.name	Required: The name of the file. For example: `svchost.exe`	string
process.file.parent_folder	Optional: The parent folder in which the file resides.	string
process.file.path	Recommended: The full path to the file. .	string
process.file.size	Optional: The size of data, in bytes.	long
process.file.type	Optional: The file type.	string
process.file.type_id	Required: The file type ID.	integer
process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer

Field	Description	Value Type
process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
process.parent_process	Recommended: The parent process of this process object.	object
process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
process.parent_ process.container	Recommended: The information describing an instance of a container.	object
process.parent_ process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e....`	object
process.parent_ process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.parent_ process.container.hash.value	Required: The digital fingerprint value.	string
process.parent_ process.container.image	Recommended: The container image used as a template to run the container.	object

Field	Description	Value Type
process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
process.parent_process.container.name	Recommended: The container name.	string
process.parent_process.container.size	Recommended: The size of the container image.	long
process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff...`.	string
process.parent_process.created_time	Recommended: The time when the process was created/started.	long
process.parent_process.file	Recommended: The process file object.	object
process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects

Field	Description	Value Type
process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string
process.parent_process.file.path	Recommended: The full path to the file. .	string
process.parent_process.file.type	Optional: The file type.	string
process.parent_process.file.type_id	Required: The file type ID.	integer
process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
process.parent_process.parent_process	Recommended: The parent process of this process object.	object

Field	Description	Value Type
process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object
process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b...`.	object
process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
process.parent_process.parent_process.container.name	Recommended: The container name.	string
process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff3....`	string
process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
process.parent_process.parent_process.file	Recommended: The process file object.	object
process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
process.parent_process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string

Field	Description	Value Type
process.parent_process.parent_process.file.path	Recommended: The full path to the file. .	string
process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
process.parent_process.parent_process.user.uid	Recommended: The unique user identifier.	string
process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
process.parent_process.session	Optional: The user session under which this process is running.	object

Field	Description	Value Type
process.parent_process.session.created_time	Recommended: The time when the session was created.	long
process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string
process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
process.parent_process.user	Recommended: The user under which this process is running.	object
process.parent_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
process.parent_process.user.type_id	Recommended: The account type identifier.	integer
process.parent_process.user.uid	Recommended: The unique user identifier.	string
process.pid	Recommended: The process identifier, as reported by the operating system.	integer
process.session	Optional: The user session under which this process is running.	object
process.session.created_time	Recommended: The time when the session was created.	long

Field	Description	Value Type
process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
process.session.issuer	Recommended: The identifier of the session issuer.	string
process.session.uid	Recommended: The unique identifier of the session.	string
process.session.uuid	Optional: The universally unique identifier of the session.	string
process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
process.user	Recommended: The user under which this process is running.	object
process.user.account	Optional: The user's account or the account associated with the user.	object
process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
process.user.account.type_id	Recommended: The normalized account type identifier.	integer
process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string

Field	Description	Value Type
process.user.email_addr	Optional: The user's primary email address.	string
process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
process.user.name	Recommended: The username. For example, `janedoe1`.	string
process.user.org	Optional: Organization and org unit related to the user.	object
process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
process.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
process.user.type_id	Recommended: The account type identifier.	integer
process.user.uid	Recommended: The unique user identifier.	string

Field	Description	Value Type
query_info	Recommended: The search details associated with the query request.	object
query_info.name	Recommended: The query name for a saved or scheduled query.	string
query_info.query_string	Required: A string representing the query code being run.	string
query_info.uid	Recommended: The unique identifier of the query.	string
query_result	Recommended: The result of the query.	string
query_result_id	Required: The normalized identifier of the query result.	integer
raw_data	Optional: The raw event/finding data as received from the source.	string
raw_data_size	Optional: The size of the raw data which was transformed into an OCSF event, in bytes.	long
service	Required: The service that pertains to the event.	object
service.name	Recommended: The name of the service.	string
service.uid	Recommended: The unique identifier of the service.	string
service.version	Recommended: The version of the service.	string
session	Required: The authenticated user or service session.	object

Field	Description	Value Type
session.created_time	Recommended: The time when the session was created.	long
session.is_remote	Recommended: The indication of whether the session is remote.	boolean
session.issuer	Recommended: The identifier of the session issuer.	string
session.uid	Recommended: The unique identifier of the session.	string
session.uuid	Optional: The universally unique identifier of the session.	string
severity	Optional: The event severity, normalized to the caption of the severity_id value.	string
severity_id	Required: <p>The normalized identifier of the event/finding severity.	integer
start_time	Optional: The start time of a time period, or the time of the least recent event included in the aggregate event.	long
state	Recommended: The state of the socket, normalized to the caption of the state_id value.	string
state_id	Required: The state of the socket.	integer
status	Recommended: The event status, normalized to the caption of the status_id value.	string
status_code	Recommended: The event status code, as reported by the event source.	string

Field	Description	Value Type
status_detail	Recommended: The status details contains additional information about the event/finding outcome.	string
status_id	Recommended: The normalized identifier of the event status.	integer
timezone_offset	Recommended: The number of minutes that the reported event `time` is ahead or behind UTC, in the range -1,080 to +1,080.	integer
type_name	Optional: The event type name, as defined by the type_uid.	string
type_uid	Required: The event/finding type ID. It identifies the event's semantics and structure.	long
unmapped	Optional: The attributes that are not mapped to the event schema.	object
user	Required: The user that is being discovered by an inventory process.	object
user.account	Optional: The user's account or the account associated with the user.	object
user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
user.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
user.email_addr	Optional: The user's primary email address.	string
user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
user.name	Recommended: The username. For example, `janedoe1`.	string
user.org	Optional: Organization and org unit related to the user.	object
user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
user.type	Optional: The type of the user. For example, System, AWS IAM User.	string

Field	Description	Value Type
user.type_id	Recommended: The account type identifier.	integer
user.uid	Recommended: The unique user identifier.	string
users	Recommended: The users that belong to the administrative group.	array of objects
users.account	Optional: The user's account or the account associated with the user.	object
users.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
users.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
users.account.type_id	Recommended: The normalized account type identifier.	integer
users.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
users.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
users.email_addr	Optional: The user's primary email address.	string
users.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
users.name	Recommended: The username. For example, `janedoe1`.	string

Field	Description	Value Type
users.org	Optional: Organization and org unit related to the user.	object
users.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
users.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
users.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
users.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
users.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
users.type_id	Recommended: The account type identifier.	integer
users.uid	Recommended: The unique user identifier.	string
volume	Required: The allocated event volume for the event in bytes.	long

OCSF Findings Schema

The Findings class captures security detections, alerts, and findings from security monitoring tools. This includes vulnerability detections, threat intelligence matches, compliance violations, anomaly detections, and security alerts from EDR, SIEM, and other security platforms. This schema is part of the OCSF (Open Cybersecurity

Schema Framework) standard. For general OCSF guidance and field naming conventions, see "OCSF Schema Overview" on page 39.

For more information on the OCSF Version 1.2 standard, see OCSF v1.2 Documentation.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Field	Description	Value Type
action	Optional: The normalized caption of 'action_id' or the source-specific action.	string
action_id	Required: The normalized action taken by a control or other policy-based system leading to an outcome or disposition.	integer
activity_id	Required: The normalized identifier of the activity that triggered the event.	integer
activity_name	Optional: The event activity name, as defined by the activity_id.	string
actor	Optional: The actor object describes details about the user/role/process that was the source of the activity.	object
actor.app_uid	Optional: The unique identifier of the client application or service that initiated the activity.	string
actor.idp	Optional: This object describes details about the Identity Provider used.	object
actor.idp.name	Recommended: The name of the identity provider.	string
actor.idp.uid	Recommended: The unique identifier of the identity provider.	string

Field	Description	Value Type
actor.invoked_by	Optional: The name of the service that invoked the activity as described in the event.	string
actor.process	Recommended: The process that initiated the activity.	object
actor.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.container	Recommended: The information describing an instance of a container.	object
actor.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b....`	object
actor.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string

Field	Description	Value Type
actor.process.container.name	Recommended: The container name.	string
actor.process.container.size	Recommended: The size of the container image.	long
actor.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc...`.	string
actor.process.created_time	Recommended: The time when the process was created/started.	long
actor.process.file	Recommended: The process file object.	object
actor.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string

Field	Description	Value Type
actor.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.file.name	Required: The name of the file.	string
actor.process.file.parent_folder	Optional: The parent folder in which the file resides.	string
actor.process.file.path	Recommended: The full path to the file. .	string
actor.process.file.size	Optional: The size of data, in bytes.	long
actor.process.file.type	Optional: The file type.	string
actor.process.file.type_id	Required: The file type ID.	integer
actor.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string

Field	Description	Value Type
actor.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.container	Recommended: The information describing an instance of a container.	object
actor.process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e...`.	object
actor.process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.container.name	Recommended: The container name.	string

Field	Description	Value Type
actor.process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc8...`	string
actor.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
actor.process.parent_process.file.name	Required: The name of the file.	string
actor.process.parent_process.file.path	Recommended: The full path to the file. .	string
actor.process.parent_process.file.type	Optional: The file type.	string
actor.process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
actor.process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5....`	object
actor.process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82f....`	string

Field	Description	Value Type
actor.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.file.name	Required: The name of the file.	string

Field	Description	Value Type
actor.process.parent_process.parent_process.file.path	Recommended: The full path to the file. .	string
actor.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
actor.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.session	Optional: The user session under which this process is running.	object
actor.process.parent_process.session.created_time	Recommended: The time when the session was created.	long
actor.process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
actor.process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.session	Optional: The user session under which this process is running.	object
actor.process.session.created_time	Recommended: The time when the session was created.	long
actor.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.session.uuid	Optional: The universally unique identifier of the session.	string
actor.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.user	Recommended: The user under which this process is running.	object
actor.process.user.account	Optional: The user's account or the account associated with the user.	object
actor.process.user.account.name	Recommended: The name of the account (for example GCP Account Name).	string
actor.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.process.user.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
actor.process.user.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
actor.process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.user.email_addr	Optional: The user's primary email address.	string
actor.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.user.org	Optional: Organization and org unit related to the user.	object
actor.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.process.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string

Field	Description	Value Type
actor.process.user.type_id	Recommended: The account type identifier.	integer
actor.process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
actor.session	Optional: The user session from which the activity was initiated.	object
actor.session.created_time	Recommended: The time when the session was created.	long
actor.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.session.issuer	Recommended: The identifier of the session issuer.	string
actor.session.uid	Recommended: The unique identifier of the session.	string
actor.session.uuid	Optional: The universally unique identifier of the session.	string
actor.user	Recommended: The user that initiated the activity or the user context from which the activity was initiated.	object
actor.user.account	Optional: The user's account or the account associated with the user.	object
actor.user.account.name	Recommended: The name of the account (for example GCP Account Name).	string
actor.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string

Field	Description	Value Type
actor.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.user.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
actor.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.user.email_addr	Optional: The user's primary email address.	string
actor.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.user.org	Optional: Organization and org unit related to the user.	object
actor.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string

Field	Description	Value Type
actor.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string
actor.user.type_id	Recommended: The account type identifier.	integer
actor.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
api	Optional: Describes details about a typical API (Application Programming Interface) call.	object
api.operation	Required: Verb/Operation associated with the request	string
api.request	Recommended: Details pertaining to the API request.	object
api.request.uid	Required: The unique request identifier.	string
api.response	Recommended: Details pertaining to the API response.	object
api.response.code	Recommended: The numeric response sent to a request.	integer
api.response.error	Recommended: Error Code	string
api.response.error_message	Recommended: Error Message	string
api.response.message	Recommended: The description of the event/finding, as defined by the source.	string

Field	Description	Value Type
api.service	Optional: The information pertaining to the API service.	object
api.service.name	Recommended: The name of the service.	string
api.service.uid	Recommended: The unique identifier of the service.	string
api.service.version	Recommended: The version of the service.	string
category_name	Optional: The event category name, as defined by category_uid value.	string
category_uid	Required: The category unique identifier of the event.	integer
class_name	Optional: The event class name, as defined by the class_uid.	string
class_uid	Required: The unique identifier of a class.	integer
cloud	Required: Describes details about the Cloud environment where the event was originally created or logged.	object
cloud.account	Optional: The account object describes details about the account that was the source or target of the activity.	object
cloud.account.name	Recommended: The name of the account (for example GCP Account Name).	string
cloud.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string

Field	Description	Value Type
cloud.account.type_id	Recommended: The normalized account type identifier.	integer
cloud.account.uid	Recommended: The unique identifier of the account.	string
cloud.org	Optional: Organization and org unit relevant to the event or object.	object
cloud.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
cloud.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
cloud.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
cloud.provider	Required: The unique name of the Cloud services provider, such as AWS, MS Azure, GCP,	string
cloud.region	Recommended: The name of the cloud region, as defined by the cloud provider.	string
compliance	Required: The compliance object provides context to compliance findings and contains compliance related details.	object
compliance.control	Recommended: A Control is prescriptive, prioritized, and simplified set of best practices that one can use to strengthen their cybersecurity posture.	string

Field	Description	Value Type
compliance.standards	Required: Security standards are a set of criteria organizations can follow to protect sensitive and confidential information.	array of strings
compliance.status	Recommended: The resultant status of the compliance check normalized to the caption of the `status_id` value.	string
compliance.status_id	Recommended: The normalized status identifier of the compliance check.	integer
confidence	Optional: The confidence, normalized to the caption of the confidence_id value. In the case of 'Other', it is defined by the event source.	string
confidence_id	Recommended: The normalized confidence refers to the accuracy of the rule that created the finding.	integer
count	Optional: The number of times that events in the same logical group occurred during the event Start Time to End Time period.	integer
data_security	Recommended: The Data Security object describes the characteristics, techniques and content of a Data Loss Prevention (DLP), Data Loss Detection (DLD), Data Classification, or similar tools' finding, alert, or detection mechanism(s).	object
data_security.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
data_security.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
data_security.data_lifecycle_state_id	Recommended: The stage or state that the data was in when it was assessed or scanned by a data security tool.	integer
data_security.detection_pattern	Recommended: Specific pattern, algorithm, fingerpint, or model used for detection.	string
data_security.detection_system_id	Recommended: The type of data security tool or system that the finding, detection, or alert originated from.	integer
data_security.policy	Recommended: Details about the policy that triggered the finding.	object
data_security.policy.name	Recommended: The policy name. For example: `IAM Policy`.	string
data_security.policy.uid	Recommended: A unique identifier of the policy instance.	string
data_security.policy.version	Recommended: The policy version number.	string
database	Recommended: Describes the database where classified or sensitive data is stored in, or was accessed from.	object
database.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
database.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
database.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
database.name	Recommended: The database name, ordinarily as assigned by a database administrator.	string
database.type	Recommended: The database type.	string
database.type_id	Required: The normalized identifier of the database type.	integer
database.uid	Recommended: The unique identifier of the database.	string
databucket	Recommended: Describes the databucket where classified or sensitive data is stored in, or was accessed from.	object
databucket.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
databucket.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
databucket.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
databucket.name	Recommended: The databucket name.	string
databucket.type	Recommended: The databucket type.	string

Field	Description	Value Type
databucket.type_id	Required: The normalized identifier of the databucket type.	integer
databucket.uid	Recommended: The unique identifier of the databucket.	string
desc	Recommended: The short description of the Incident.	string
device	Recommended: An addressable device, computer system or host.	object
device.container	Recommended: The information describing an instance of a container.	object
device.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b....`	object
device.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
device.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
device.container.hash.value	Required: The digital fingerprint value.	string
device.container.image	Recommended: The container image used as a template to run the container.	object
device.container.image.name	Recommended: The image name. For example: `elixir`.	string
device.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string

Field	Description	Value Type
device.container.name	Recommended: The container name.	string
device.container.size	Recommended: The size of the container image.	long
device.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82....`	string
device.domain	Optional: The network domain where the device resides. For example: `work.example.com`.	string
device.groups	Optional: The group names to which the device belongs. For example: `["Windows Laptops", "Engineering"]`.	array of objects
device.groups.desc	Optional: The group description.	string
device.groups.domain	Optional: The domain where the group is defined. For example: the LDAP or Active Directory domain.	string
device.groups.name	Recommended: The group name.	string
device.groups.type	Optional: The type of the group or account.	string
device.groups.uid	Recommended: The unique identifier of the group.	string
device.hostname	Recommended: The device hostname.	string

Field	Description	Value Type
device.instance_uid	Recommended: The unique identifier of a VM instance.	string
device.interface_name	Recommended: The name of the network interface (for example eth2).	string
device.interface_uid	Recommended: The unique identifier of the network interface.	string
device.ip	Recommended: The device IP address, in either IPv4 or IPv6 format.	string
device.is_compliant	Optional: The event occurred on a compliant device.	boolean
device.is_managed	Optional: The event occurred on a managed device.	boolean
device.name	Recommended: The alternate device name, ordinarily as assigned by an administrator.	string
device.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
device.os	Optional: The endpoint operating system.	object
device.os.name	Required: The operating system name.	string
device.os.type	Optional: The type of the operating system.	string
device.os.type_id	Required: The type identifier of the operating system.	integer

Field	Description	Value Type
device.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
device.owner.account	Optional: The user's account or the account associated with the user.	object
device.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
device.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
device.owner.account.type_id	Recommended: The normalized account type identifier.	integer
device.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
device.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
device.owner.email_addr	Optional: The user's primary email address.	string
device.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
device.owner.name	Recommended: The username. For example, `janedoe1`.	string
device.owner.org	Optional: Organization and org unit related to the user.	object
device.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string

Field	Description	Value Type
device.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
device.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
device.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
device.owner.type	Optional: The type of the user. For example, System, AWS IAM User.	string
device.owner.type_id	Recommended: The account type identifier.	integer
device.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
device.region	Recommended: The region where the virtual machine is located. For example, an AWS Region.	string
device.type	Optional: The device type. For example: `unknown`, `server`, `desktop`, `laptop`, `tablet`, `mobile`, `virtual`, `browser`, or `other`.	string
device.type_id	Required: The device type ID.	integer
device.uid	Recommended: The unique identifier of the device. For example the Windows TargetSID or AWS EC2 ARN.	string

Field	Description	Value Type
dst_endpoint	Recommended: Describes the endpoint where classified or sensitive data is stored in, or was accessed from.	object
dst_endpoint.container	Recommended: The information describing an instance of a container.	object
dst_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b440....`	object
dst_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
dst_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
dst_endpoint.container.name	Recommended: The container name.	string
dst_endpoint.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
dst_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff3...`.	string
dst_endpoint.domain	Optional: The name of the domain.	string
dst_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
dst_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
dst_endpoint.interface_name	Recommended: The name of the network interface.	string
dst_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.location	Optional: The geographical location of the endpoint.	object
dst_endpoint.location.city	Recommended: The name of the city.	string
dst_endpoint.location.continent	Recommended: The name of the continent.	string
dst_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
dst_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: `42.361145`.	float

Field	Description	Value Type
dst_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: `-71.057083`.	float
dst_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (for example province or state) of the country.	string
dst_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
dst_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
dst_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
dst_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
dst_endpoint.owner.account.name	Recommended: The name of the account (for example, GCP Account Name).	string
dst_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
dst_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
dst_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
dst_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
dst_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
dst_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
dst_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
dst_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string

Field	Description	Value Type
dst_endpoint.owner.type_id	Recommended: The account type identifier.	integer
dst_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
dst_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
dst_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object
dst_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b4....`	object
dst_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string

Field	Description	Value Type
dst_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
dst_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
dst_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
dst_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string
dst_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
dst_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
dst_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (for example eth2).	string
dst_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
dst_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object

Field	Description	Value Type
dst_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
dst_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
dst_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
dst_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
dst_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
dst_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
dst_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
dst_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string

Field	Description	Value Type
dst_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
dst_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
dst_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer
dst_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
dst_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
dst_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string

Field	Description	Value Type
dst_endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.zone	Optional: The network zone or LAN segment.	string
duration	Optional: The event duration or aggregate time, the amount of time the event covers from `start_time` to `end_time` in milliseconds.	integer
end_time	Optional: The end time of a time period, or the time of the most recent event included in the aggregate event.	long
event_seq_id	Required: The sequence identifier for the event.	long
event_size	Required: The size of the event in bytes.	long
event_time	Required: The normalized event occurrence time or the finding creation time.	long
evidences	Recommended: Describes various evidence artifacts associated to the activity/activities that triggered a security detection.	array of objects
evidences.actor	Recommended: Describes details about the user/role/process that was the source of the activity that triggered the detection.	object

Field	Description	Value Type
evidences.actor.app_uid	Optional: The unique identifier of the client application or service that initiated the activity.	string
evidences.actor.idp	Optional: This object describes details about the Identity Provider used.	object
evidences.actor.idp.name	Recommended: The name of the identity provider.	string
evidences.actor.idp.uid	Recommended: The unique identifier of the identity provider.	string
evidences.actor.invoked_by	Optional: The name of the service that invoked the activity as described in the event.	string
evidences.actor.process	Recommended: The process that initiated the activity.	object
evidences.actor.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
evidences.actor.process.container	Recommended: The information describing an instance of a container.	object
evidences.actor.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b....`	object
evidences.actor.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.actor.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
evidences.actor.process.container.hash.value	Required: The digital fingerprint value.	string
evidences.actor.process.container.image	Recommended: The container image used as a template to run the container.	object
evidences.actor.process.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.actor.process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.actor.process.container.name	Recommended: The container name.	string
evidences.actor.process.container.size	Recommended: The size of the container image.	long
evidences.actor.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff...`	string
evidences.actor.process.created_time	Recommended: The time when the process was created/started.	long
evidences.actor.process.file	Recommended: The process file object.	object
evidences.actor.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.actor.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
evidences.actor.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.actor.process.file.hashes	Recommended: An array of hash attributes.	array of objects
evidences.actor.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.actor.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.actor.process.file.hashes.value	Required: The digital fingerprint value.	string
evidences.actor.process.file.name	Required: The name of the file.	string
evidences.actor.process.file.parent_folder	Optional: The parent folder in which the file resides.	string
evidences.actor.process.file.path	Recommended: The full path to the file. .	string
evidences.actor.process.file.size	Optional: The size of data, in bytes.	long
evidences.actor.process.file.type	Optional: The file type.	string
evidences.actor.process.file.type_id	Required: The file type ID.	integer

Field	Description	Value Type
evidences.actor.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
evidences.actor.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
evidences.actor.process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
evidences.actor.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.actor.process.parent_process	Recommended: The parent process of this process object.	object
evidences.actor.process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
evidences.actor.process.parent_process.container	Recommended: The information describing an instance of a container.	object
evidences.actor.process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5....`	object
evidences.actor.process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
evidences.actor.process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
evidences.actor.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
evidences.actor.process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.actor.process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.actor.process.parent_process.container.name	Recommended: The container name.	string
evidences.actor.process.parent_process.container.size	Recommended: The size of the container image.	long
evidences.actor.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string
evidences.actor.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
evidences.actor.process.parent_process.file	Recommended: The process file object.	object
evidences.actor.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.actor.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
evidences.actor.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.actor.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
evidences.actor.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.actor.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
evidences.actor.process.parent_process.file.name	Required: The name of the file.	string
evidences.actor.process.parent_process.file.path	Recommended: The full path to the file. .	string
evidences.actor.process.parent_process.file.type	Optional: The file type.	string
evidences.actor.process.parent_process.file.type_id	Required: The file type ID.	integer
evidences.actor.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
evidences.actor.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer

Field	Description	Value Type
evidences.actor.process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
evidences.actor.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.actor.process.parent_process.parent_process	Recommended: The parent process of this process object.	object
evidences.actor.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
evidences.actor.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object
evidences.actor.process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b4....`	object
evidences.actor.process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.actor.process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
evidences.actor.process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object

Field	Description	Value Type
evidences.actor.process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.actor.process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.actor.process.parent_process.parent_process.container.name	Recommended: The container name.	string
evidences.actor.process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
evidences.actor.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff...`.	string
evidences.actor.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
evidences.actor.process.parent_process.parent_process.file	Recommended: The process file object.	object
evidences.actor.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.actor.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
evidences.actor.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.actor.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
evidences.actor.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.actor.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
evidences.actor.process.parent_process.parent_process.file.name	Required: The name of the file.	string
evidences.actor.process.parent_process.parent_process.file.path	Recommended: The full path to the file.	string
evidences.actor.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
evidences.actor.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
evidences.actor.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
evidences.actor.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
evidences.actor.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
evidences.actor.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
evidences.actor.process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
evidences.actor.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
evidences.actor.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.actor.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
evidences.actor.process.parent_process.session	Optional: The user session under which this process is running.	object
evidences.actor.process.parent_process.session.created_time	Recommended: The time when the session was created.	long
evidences.actor.process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
evidences.actor.process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string

Field	Description	Value Type
evidences.actor.process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
evidences.actor.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
evidences.actor.process.parent_process.user	Recommended: The user under which this process is running.	object
evidences.actor.process.parent_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.actor.process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
evidences.actor.process.parent_process.user.type_id	Recommended: The account type identifier.	integer
evidences.actor.process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.actor.process.pid	Recommended: The process identifier, as reported by the operating system.	integer
evidences.actor.process.session	Optional: The user session under which this process is running.	object
evidences.actor.process.session.created_time	Recommended: The time when the session was created.	long
evidences.actor.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
evidences.actor.process.session.issuer	Recommended: The identifier of the session issuer.	string

Field	Description	Value Type
evidences.actor.process.session.uid	Recommended: The unique identifier of the session.	string
evidences.actor.process.session.uuid	Optional: The universally unique identifier of the session.	string
evidences.actor.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
evidences.actor.process.user	Recommended: The user under which this process is running.	object
evidences.actor.process.user.account	Optional: The user's account or the account associated with the user.	object
evidences.actor.process.user.account.name	Recommended: The name of the account (for example GCP Account Name).	string
evidences.actor.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
evidences.actor.process.user.account.type_id	Recommended: The normalized account type identifier.	integer
evidences.actor.process.user.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
evidences.actor.process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.actor.process.user.email_addr	Optional: The user's primary email address.	string
evidences.actor.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
evidences.actor.process.user.name	Recommended: The username. For example, `janedoe1`.	string
evidences.actor.process.user.org	Optional: Organization and org unit related to the user.	object
evidences.actor.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
evidences.actor.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
evidences.actor.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
evidences.actor.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
evidences.actor.process.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string
evidences.actor.process.user.type_id	Recommended: The account type identifier.	integer
evidences.actor.process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.actor.session	Optional: The user session from which the activity was initiated.	object
evidences.actor.session.created_time	Recommended: The time when the session was created.	long
evidences.actor.session.is_remote	Recommended: The indication of whether the session is remote.	boolean

Field	Description	Value Type
evidences.actor.session.issuer	Recommended: The identifier of the session issuer.	string
evidences.actor.session.uid	Recommended: The unique identifier of the session.	string
evidences.actor.session.uuid	Optional: The universally unique identifier of the session.	string
evidences.actor.user	Recommended: The user that initiated the activity or the user context from which the activity was initiated.	object
evidences.actor.user.account	Optional: The user's account or the account associated with the user.	object
evidences.actor.user.account.name	Recommended: The name of the account (for example GCP Account Name).	string
evidences.actor.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
evidences.actor.user.account.type_id	Recommended: The normalized account type identifier.	integer
evidences.actor.user.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
evidences.actor.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.actor.user.email_addr	Optional: The user's primary email address.	string
evidences.actor.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
evidences.actor.user.name	Recommended: The username. For example, `janedoe1`.	string
evidences.actor.user.org	Optional: Organization and org unit related to the user.	object
evidences.actor.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
evidences.actor.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
evidences.actor.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
evidences.actor.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
evidences.actor.user.type	Optional: The type of the user. For example, System, AWS IAM User.	string
evidences.actor.user.type_id	Recommended: The account type identifier.	integer
evidences.actor.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.api	Recommended: Describes details about the API call associated to the activity that triggered the detection.	object
evidences.api.operation	Required: Verb/Operation associated with the request	string

Field	Description	Value Type
evidences.api.request	Recommended: Details pertaining to the API request.	object
evidences.api.request.uid	Required: The unique request identifier.	string
evidences.api.response	Recommended: Details pertaining to the API response.	object
evidences.api.response.code	Recommended: The numeric response sent to a request.	integer
evidences.api.response.error	Recommended: Error Code	string
evidences.api.response.error_message	Recommended: Error Message	string
evidences.api.response.message	Recommended: The description of the event/finding, as defined by the source.	string
evidences.api.service	Optional: The information pertaining to the API service.	object
evidences.api.service.name	Recommended: The name of the service.	string
evidences.api.service.uid	Recommended: The unique identifier of the service.	string
evidences.api.service.version	Recommended: The version of the service.	string
evidences.connection_info	Recommended: Describes details about the network connection associated to the activity that triggered the detection.	object

Field	Description	Value Type
evidences.connection_info.direction	Optional: The direction of the initiated connection, traffic, or email, normalized to the caption of the direction_id value.	string
evidences.connection_info.direction_id	Required: The normalized identifier of the direction of the initiated connection, traffic, or email.	integer
evidences.connection_info.protocol_name	Optional: The TCP/IP protocol name in lowercase, as defined by the Internet Assigned Numbers Authority (IANA).	string
evidences.connection_info.protocol_num	Recommended: The TCP/IP protocol number, as defined by the Internet Assigned Numbers Authority (IANA). For example: `6` for TCP and `17` for UDP.	integer
evidences.connection_info.session	Optional: The authenticated user or service session.	object
evidences.connection_info.session.created_time	Recommended: The time when the session was created.	long
evidences.connection_info.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
evidences.connection_info.session.issuer	Recommended: The identifier of the session issuer.	string
evidences.connection_info.session.uid	Recommended: The unique identifier of the session.	string
evidences.connection_info.session.uuid	Optional: The universally unique identifier of the session.	string
evidences.connection_info.uid	Optional: The unique identifier of the connection.	string

Field	Description	Value Type
evidences.container	Recommended: Describes details about the container associated to the activity that triggered the detection.	object
evidences.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b44....`	object
evidences.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.container.hash.value	Required: The digital fingerprint value.	string
evidences.container.image	Recommended: The container image used as a template to run the container.	object
evidences.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.container.name	Recommended: The container name.	string
evidences.container.size	Recommended: The size of the container image.	long
evidences.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string

Field	Description	Value Type
evidences.database	Recommended: Describes details about the database associated to the activity that triggered the detection.	object
evidences.database.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.database.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
evidences.database.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.database.name	Recommended: The database name, ordinarily as assigned by a database administrator.	string
evidences.database.type	Recommended: The database type.	string
evidences.database.type_id	Required: The normalized identifier of the database type.	integer
evidences.database.uid	Recommended: The unique identifier of the database.	string
evidences.databucket	Recommended: Describes details about the databucket associated to the activity that triggered the detection.	object
evidences.databucket.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
evidences.databucket.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
evidences.databucket.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.databucket.name	Recommended: The databucket name.	string
evidences.databucket.type	Recommended: The databucket type.	string
evidences.databucket.type_id	Required: The normalized identifier of the databucket type.	integer
evidences.databucket.uid	Recommended: The unique identifier of the databucket.	string
evidences.dst_endpoint	Recommended: Describes details about the destination of the network activity that triggered the detection.	object
evidences.dst_endpoint.container	Recommended: The information describing an instance of a container.	object
evidences.dst_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b440....`	object
evidences.dst_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.dst_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
evidences.dst_endpoint.container.hash.value	Required: The digital fingerprint value.	string
evidences.dst_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
evidences.dst_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.dst_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.dst_endpoint.container.name	Recommended: The container name.	string
evidences.dst_endpoint.container.size	Recommended: The size of the container image.	long
evidences.dst_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff...`.	string
evidences.dst_endpoint.domain	Optional: The name of the domain.	string
evidences.dst_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
evidences.dst_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
evidences.dst_endpoint.interface_name	Recommended: The name of the network interface.	string
evidences.dst_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
evidences.dst_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string

Field	Description	Value Type
evidences.dst_endpoint.location	Optional: The geographical location of the endpoint.	object
evidences.dst_endpoint.location.city	Recommended: The name of the city.	string
evidences.dst_endpoint.location.continent	Recommended: The name of the continent.	string
evidences.dst_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
evidences.dst_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: `42.361145`.	float
evidences.dst_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: `-71.057083`.	float
evidences.dst_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (for example province or state) of the country.	string
evidences.dst_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
evidences.dst_endpoint.name	Recommended: The short name of the endpoint.	string
evidences.dst_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
evidences.dst_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
evidences.dst_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
evidences.dst_endpoint.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
evidences.dst_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
evidences.dst_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
evidences.dst_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
evidences.dst_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.dst_endpoint.owner.email_addr	Optional: The user's primary email address.	string
evidences.dst_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
evidences.dst_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
evidences.dst_endpoint.owner.org	Optional: Organization and org unit related to the user.	object

Field	Description	Value Type
evidences.dst_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
evidences.dst_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
evidences.dst_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
evidences.dst_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
evidences.dst_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
evidences.dst_endpoint.owner.type_id	Recommended: The account type identifier.	integer
evidences.dst_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.dst_endpoint.port	Recommended: The port used for communication within the network connection.	integer
evidences.dst_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
evidences.dst_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
evidences.dst_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b440....`	object
evidences.dst_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.dst_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.dst_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
evidences.dst_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
evidences.dst_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.dst_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.dst_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
evidences.dst_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
evidences.dst_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string
evidences.dst_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string

Field	Description	Value Type
evidences.dst_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
evidences.dst_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (for example eth2).	string
evidences.dst_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
evidences.dst_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
evidences.dst_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
evidences.dst_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.dst_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
evidences.dst_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
evidences.dst_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (for example, GCP Account Name).	string
evidences.dst_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
evidences.dst_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
evidences.dst_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example, AWS Account ID).	string
evidences.dst_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.dst_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
evidences.dst_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
evidences.dst_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
evidences.dst_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
evidences.dst_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
evidences.dst_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
evidences.dst_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
evidences.dst_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
evidences.dst_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string

Field	Description	Value Type
evidences.dst_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer
evidences.dst_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.dst_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
evidences.dst_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
evidences.dst_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
evidences.dst_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
evidences.dst_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
evidences.dst_endpoint.type_id	Recommended: The network endpoint type ID.	integer
evidences.dst_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
evidences.dst_endpoint.zone	Optional: The network zone or LAN segment.	string
evidences.file	Recommended: Describes details about the file associated to the activity that triggered the detection.	object

Field	Description	Value Type
evidences.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
evidences.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.file.hashes	Recommended: An array of hash attributes.	array of objects
evidences.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.file.hashes.value	Required: The digital fingerprint value.	string
evidences.file.name	Required: The name of the file.	string
evidences.file.parent_folder	Optional: The parent folder in which the file resides.	string
evidences.file.path	Recommended: The full path to the file. .	string
evidences.file.size	Optional: The size of data, in bytes.	long

Field	Description	Value Type
evidences.file.type	Optional: The file type.	string
evidences.file.type_id	Required: The file type ID.	integer
evidences.process	Recommended: Describes details about the process associated to the activity that triggered the detection.	object
evidences.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
evidences.process.container	Recommended: The information describing an instance of a container.	object
evidences.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b4...`.	object
evidences.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.process.container.hash.value	Required: The digital fingerprint value.	string
evidences.process.container.image	Recommended: The container image used as a template to run the container.	object
evidences.process.container.image.name	Recommended: The image name. For example: `elixir`.	string

Field	Description	Value Type
evidences.process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.process.container.name	Recommended: The container name.	string
evidences.process.container.size	Recommended: The size of the container image.	long
evidences.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string
evidences.process.created_time	Recommended: The time when the process was created/started.	long
evidences.process.file	Recommended: The process file object.	object
evidences.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
evidences.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.process.file.hashes	Recommended: An array of hash attributes.	array of objects

Field	Description	Value Type
evidences.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.process.file.hashes.value	Required: The digital fingerprint value.	string
evidences.process.file.name	Required: The name of the file.	string
evidences.process.file.parent_folder	Optional: The parent folder in which the file resides.	string
evidences.process.file.path	Recommended: The full path to the file. .	string
evidences.process.file.size	Optional: The size of data, in bytes.	long
evidences.process.file.type	Optional: The file type.	string
evidences.process.file.type_id	Required: The file type ID.	integer
evidences.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
evidences.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
evidences.process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string

Field	Description	Value Type
evidences.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.process.parent_process	Recommended: The parent process of this process object.	object
evidences.process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
evidences.process.parent_process.container	Recommended: The information describing an instance of a container.	object
evidences.process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b4...`.	object
evidences.process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
evidences.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
evidences.process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.process.parent_process.container.name	Recommended: The container name.	string

Field	Description	Value Type
evidences.process.parent_process.container.size	Recommended: The size of the container image.	long
evidences.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc82ff....`	string
evidences.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
evidences.process.parent_process.file	Recommended: The process file object.	object
evidences.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
evidences.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
evidences.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
evidences.process.parent_process.file.name	Required: The name of the file.	string
evidences.process.parent_process.file.path	Recommended: The full path to the file. .	string
evidences.process.parent_process.file.type	Optional: The file type.	string
evidences.process.parent_process.file.type_id	Required: The file type ID.	integer
evidences.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
evidences.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
evidences.process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
evidences.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.process.parent_process.parent_process	Recommended: The parent process of this process object.	object
evidences.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
evidences.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
evidences.process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b440161c2b33a3e4f302ac....`	object
evidences.process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
evidences.process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
evidences.process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.process.parent_process.parent_process.container.name	Recommended: The container name.	string
evidences.process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
evidences.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `13550340a8681c84c861aa....`	string

Field	Description	Value Type
evidences.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
evidences.process.parent_process.parent_process.file	Recommended: The process file object.	object
evidences.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
evidences.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
evidences.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
evidences.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
evidences.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
evidences.process.parent_process.parent_process.file.name	Required: The name of the file.	string

Field	Description	Value Type
evidences.process.parent_process.parent_process.file.path	Recommended: The full path to the file.	string
evidences.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
evidences.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
evidences.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
evidences.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
evidences.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
evidences.process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
evidences.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
evidences.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
evidences.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
evidences.process.parent_process.session	Optional: The user session under which this process is running.	object
evidences.process.parent_process.session.created_time	Recommended: The time when the session was created.	long
evidences.process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
evidences.process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string
evidences.process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
evidences.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
evidences.process.parent_process.user	Recommended: The user under which this process is running.	object
evidences.process.parent_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
evidences.process.parent_process.user.type_id	Recommended: The account type identifier.	integer
evidences.process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
evidences.process.pid	Recommended: The process identifier, as reported by the operating system.	integer
evidences.process.session	Optional: The user session under which this process is running.	object
evidences.process.session.created_time	Recommended: The time when the session was created.	long
evidences.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
evidences.process.session.issuer	Recommended: The identifier of the session issuer.	string
evidences.process.session.uid	Recommended: The unique identifier of the session.	string
evidences.process.session.uuid	Optional: The universally unique identifier of the session.	string
evidences.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
evidences.process.user	Recommended: The user under which this process is running.	object
evidences.process.user.account	Optional: The user's account or the account associated with the user.	object
evidences.process.user.account.name	Recommended: The name of the account (for example GCP Account Name).	string
evidences.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
evidences.process.user.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
evidences.process.user.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
evidences.process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.process.user.email_addr	Optional: The user's primary email address.	string
evidences.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
evidences.process.user.name	Recommended: The username. For example, `janedoe1`.	string
evidences.process.user.org	Optional: Organization and org unit related to the user.	object
evidences.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
evidences.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
evidences.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
evidences.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
evidences.process.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string

Field	Description	Value Type
evidences.process.user.type_id	Recommended: The account type identifier.	integer
evidences.process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.query	Recommended: Describes details about the DNS query associated to the activity that triggered the detection.	object
evidences.query.class	Recommended: The class of resource records being queried.	string
evidences.query.hostname	Required: The hostname or domain being queried. For example: `www.example.com`	string
evidences.query.opcode_id	Recommended: The DNS opcode ID specifies the normalized query message type.	integer
evidences.query.packet_uid	Recommended: The DNS packet identifier assigned by the program that generated the query.	integer
evidences.query.type	Recommended: The type of resource records being queried.	string
evidences.src_endpoint	Recommended: Describes details about the source of the network activity that triggered the detection.	object
evidences.src_endpoint.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
evidences.src_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c86....`	object
evidences.src_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.src_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.src_endpoint.container.hash.value	Required: The digital fingerprint value.	string
evidences.src_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
evidences.src_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
evidences.src_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.src_endpoint.container.name	Recommended: The container name.	string
evidences.src_endpoint.container.size	Recommended: The size of the container image.	long
evidences.src_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `13550340a8681c84c861aa....`	string
evidences.src_endpoint.domain	Optional: The name of the domain.	string

Field	Description	Value Type
evidences.src_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
evidences.src_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
evidences.src_endpoint.interface_name	Recommended: The name of the network interface (for example eth2).	string
evidences.src_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
evidences.src_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
evidences.src_endpoint.location	Optional: The geographical location of the endpoint.	object
evidences.src_endpoint.location.city	Recommended: The name of the city.	string
evidences.src_endpoint.location.continent	Recommended: The name of the continent.	string
evidences.src_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
evidences.src_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: `42.361145`.	float
evidences.src_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: `-71.057083`.	float

Field	Description	Value Type
evidences.src_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (for example province or state) of the country.	string
evidences.src_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
evidences.src_endpoint.name	Recommended: The short name of the endpoint.	string
evidences.src_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.src_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
evidences.src_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
evidences.src_endpoint.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
evidences.src_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
evidences.src_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
evidences.src_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string

Field	Description	Value Type
evidences.src_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.src_endpoint.owner.email_addr	Optional: The user's primary email address.	string
evidences.src_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
evidences.src_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
evidences.src_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
evidences.src_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
evidences.src_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
evidences.src_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
evidences.src_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
evidences.src_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
evidences.src_endpoint.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
evidences.src_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.src_endpoint.port	Recommended: The port used for communication within the network connection.	integer
evidences.src_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
evidences.src_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object
evidences.src_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b440....`	object
evidences.src_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
evidences.src_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
evidences.src_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
evidences.src_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
evidences.src_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string

Field	Description	Value Type
evidences.src_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
evidences.src_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
evidences.src_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
evidences.src_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `13550340a8681c84c861aa...`.	string
evidences.src_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
evidences.src_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
evidences.src_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (for example eth2).	string
evidences.src_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
evidences.src_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
evidences.src_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
evidences.src_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
evidences.src_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object

Field	Description	Value Type
evidences.src_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
evidences.src_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
evidences.src_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
evidences.src_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
evidences.src_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
evidences.src_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
evidences.src_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
evidences.src_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
evidences.src_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
evidences.src_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
evidences.src_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string

Field	Description	Value Type
evidences.src_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
evidences.src_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
evidences.src_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
evidences.src_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
evidences.src_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer
evidences.src_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
evidences.src_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
evidences.src_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
evidences.src_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
evidences.src_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
evidences.src_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string

Field	Description	Value Type
evidences.src_endpoint.type_id	Recommended: The network endpoint type ID.	integer
evidences.src_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
evidences.src_endpoint.zone	Optional: The network zone or LAN segment.	string
file	Recommended: Describes a file that contains classified or sensitive data.	object
file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
file.hashes	Recommended: An array of hash attributes.	array of objects
file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
file.hashes.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
file.name	Required: The name of the file.	string
file.parent_folder	Optional: The parent folder in which the file resides. For example: `c:\windows\system32`	string
file.path	Recommended: The full path to the file.	string
file.size	Optional: The size of data, in bytes.	long
file.type	Optional: The file type.	string
file.type_id	Required: The file type ID.	integer
finding_info	Required: Describes the supporting information about a generated finding.	object
finding_info.analytic	Recommended: The analytic technique used to analyze and derive insights from the data or information that led to the finding or conclusion.	object
finding_info.analytic.name	Recommended: The name of the analytic that generated the finding.	string
finding_info.analytic.type_id	Required: The analytic type ID.	integer
finding_info.analytic.uid	Recommended: The unique identifier of the analytic that generated the finding.	string
finding_info.desc	Optional: The description of the reported finding.	string
finding_info.product_uid	Optional: The unique identifier of the product that reported the finding.	string

Field	Description	Value Type
finding_info.title	Required: A title or a brief phrase summarizing the reported finding.	string
finding_info.types	Optional: One or more types of the reported finding.	array of strings
finding_info.uid	Required: The unique identifier of the reported finding.	string
finding_info_list	Required: A list of `finding_info` objects associated to an incident.	array of objects
finding_info_list.analytic	Recommended: The analytic technique used to analyze and derive insights from the data or information that led to the finding or conclusion.	object
finding_info_list.analytic.name	Recommended: The name of the analytic that generated the finding.	string
finding_info_list.analytic.type_id	Required: The analytic type ID.	integer
finding_info_list.analytic.uid	Recommended: The unique identifier of the analytic that generated the finding.	string
finding_info_list.desc	Optional: The description of the reported finding.	string
finding_info_list.product_uid	Optional: The unique identifier of the product that reported the finding.	string
finding_info_list.title	Required: A title or a brief phrase summarizing the reported finding.	string

Field	Description	Value Type
finding_info_list.types	Optional: One or more types of the reported finding.	array of strings
finding_info_list.uid	Required: The unique identifier of the reported finding.	string
impact	Recommended: The impact , normalized to the caption of the impact_id value.	string
impact_id	Recommended: The normalized impact of the finding.	integer
impact_score	Recommended: The impact of the finding, valid range 0-100.	integer
indexed_time	Required: The time when the event was indexed.	long
message	Recommended: The description of the event/finding, as defined by the source.	string
metadata	Required: The metadata associated with the event or a finding.	object
metadata.correlation_uid	Optional: The unique identifier used to correlate events.	string
metadata.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
metadata.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.event_code	Optional: The Event ID or Code that the product uses to describe the event.	string
metadata.log_level	Optional: The audit level at which an event was generated.	string
metadata.log_name	Recommended: The event log name. For example, syslog file name or Windows logging subsystem: Security.	string
metadata.log_provider	Recommended: The logging provider or logging service that logged the event. For example, Microsoft-Windows-Security-Auditing.	string
metadata.original_time	Recommended: The original event time as reported by the event source.	string
metadata.product	Required: The product that reported the event.	object
metadata.product.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.product.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.product.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.product.feature	Optional: The feature that reported the event.	object

Field	Description	Value Type
metadata.product.feature.name	Recommended: The name of the feature.	string
metadata.product.feature.uid	Recommended: The unique identifier of the feature.	string
metadata.product.feature.version	Recommended: The version of the feature.	string
metadata.product.name	Recommended: The name of the product.	string
metadata.product.uid	Recommended: The unique identifier of the product.	string
metadata.product.vendor_name	Required: The name of the vendor of the product.	string
metadata.product.version	Recommended: The version of the product, as defined by the event source. For example: `2013.1.3-beta`.	string
metadata.profiles	Optional: The list of profiles used to create the event.	array of strings
metadata.sequence	Optional: Sequence number of the event.	integer
metadata.tenant_uid	Recommended: The unique tenant identifier.	string
metadata.uid	Optional: The logging system-assigned unique identifier of an event instance.	string
metadata.version	Required: The version of the OCSF schema.	string

Field	Description	Value Type
observables	Recommended: The observables associated with the event or a finding.	array of objects
observables.name	Required: The full name of the observable attribute. The `name` is a pointer/reference to an attribute within the event data. For example: `file.name`.	string
observables.type	Optional: The observable value type name.	string
observables.type_id	Required: The observable value type identifier.	integer
observables.value	Optional: The value associated with the observable attribute. The meaning of the value depends on the observable type.	string
original_event_time	Required: The original event time from the source system.	long
priority_id	Recommended: The normalized priority.	integer
raw_data	Optional: The raw event/finding data as received from the source.	string
raw_data_size	Optional: The size of the raw data which was transformed into an OCSF event, in bytes.	long
remediation	Recommended: Describes the recommended remediation steps to address identified issue(s).	object

Field	Description	Value Type
remediation.desc	Required: The description of the remediation strategy.	string
resource	Recommended: Describes details about the resource that is the subject of the compliance check.	object
resource.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
resource.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
resource.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
resource.owner	Recommended: The identity of the service or user account that owns the resource.	object
resource.owner.account	Optional: The user's account or the account associated with the user.	object
resource.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
resource.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
resource.owner.account.type_id	Recommended: The normalized account type identifier.	integer
resource.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string

Field	Description	Value Type
resource.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
resource.owner.email_addr	Optional: The user's primary email address.	string
resource.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
resource.owner.name	Recommended: The username. For example, `janedoe1`.	string
resource.owner.org	Optional: Organization and org unit related to the user.	object
resource.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
resource.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
resource.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
resource.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
resource.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
resource.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
resource.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
resources	Recommended: Describes details about resources that were the target of the activity that triggered the finding.	array of objects
resources.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
resources.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
resources.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
resources.owner	Recommended: The identity of the service or user account that owns the resource.	object
resources.owner.account	Optional: The user's account or the account associated with the user.	object
resources.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
resources.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
resources.owner.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
resources.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
resources.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
resources.owner.email_addr	Optional: The user's primary email address.	string
resources.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
resources.owner.name	Recommended: The username. For example, `janedoe1`.	string
resources.owner.org	Optional: Organization and org unit related to the user.	object
resources.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
resources.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
resources.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
resources.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
resources.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string

Field	Description	Value Type
resources.owner.type_id	Recommended: The account type identifier.	integer
resources.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
severity	Optional: The event severity, normalized to the caption of the severity_id value.	string
severity_id	Required: The normalized identifier of the event/finding severity. The normalized severity is a measurement the effort and expense required to manage and resolve an event or incident. Smaller numerical values represent lower impact events, and larger numerical values represent higher impact events.	integer
src_endpoint	Recommended: Details about the source endpoint where classified or sensitive data was accessed from.	object
src_endpoint.container	Recommended: The information describing an instance of a container.	object
src_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b440161c2b33a3e4f302ac....`	object
src_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string

Field	Description	Value Type
src_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
src_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
src_endpoint.container.name	Recommended: The container name.	string
src_endpoint.container.size	Recommended: The size of the container image.	long
src_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `13550340a8681c84c861aa....`	string
src_endpoint.domain	Optional: The name of the domain.	string
src_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
src_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.interface_name	Recommended: The name of the network interface (for example eth2).	string

Field	Description	Value Type
src_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.location	Optional: The geographical location of the endpoint.	object
src_endpoint.location.city	Recommended: The name of the city.	string
src_endpoint.location.continent	Recommended: The name of the continent.	string
src_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
src_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: `42.361145`.	float
src_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: `-71.057083`.	float
src_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (for example province or state) of the country.	string
src_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
src_endpoint.name	Recommended: The short name of the endpoint.	string

Field	Description	Value Type
src_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
src_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_endpoint.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
src_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
src_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
src_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.owner.email_addr	Optional: The user's primary email address.	string
src_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
src_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string

Field	Description	Value Type
src_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
src_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
src_endpoint.owner.type_id	Recommended: The account type identifier.	integer
src_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
src_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
src_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
src_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861aac2e5b440161c2b33a3e4f302ac...`.	object
src_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
src_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
src_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
src_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
src_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
src_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `13550340a8681c84c861aa...`.	string
src_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string

Field	Description	Value Type
src_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (for example eth2).	string
src_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
src_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
src_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (for example GCP Account Name).	string
src_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
src_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (for example AWS Account ID).	string
src_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
src_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
src_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
src_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
src_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string

Field	Description	Value Type
src_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer
src_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
src_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.zone	Optional: The network zone or LAN segment.	string
src_url	Recommended: A Url link used to access the original incident.	string
start_time	Optional: The start time of a time period, or the time of the least recent event included in the aggregate event.	long

Field	Description	Value Type
status	Recommended: The event status, normalized to the caption of the status_id value.	string
status_code	Recommended: The event status code, as reported by the event source.	string
status_detail	Recommended: The status details contains additional information about the event/finding outcome.	string
status_id	Recommended: The normalized identifier of the event status.	integer
table	Recommended: Describes the table where classified or sensitive data is stored in, or was accessed from.	object
table.name	Recommended: The table name, ordinarily as assigned by a database administrator.	string
table.uid	Recommended: The unique identifier of the table.	string
timezone_offset	Recommended: The number of minutes that the reported event `time` is ahead or behind UTC, in the range -1,080 to +1,080.	integer
type_name	Optional: The event type name, as defined by the type_uid.	string
type_uid	Required: The event/finding type ID. It identifies the event's semantics and structure.	long
unmapped	Optional: The attributes that are not mapped to the event schema.	object

Field	Description	Value Type
verdict	Recommended: The verdict assigned to an Incident finding.	string
verdict_id	Recommended: The normalized verdict of an Incident.	integer
volume	Required: The allocated event volume for the event in bytes.	long
vulnerabilities	Required: This object describes vulnerabilities reported in a security finding.	array of objects
vulnerabilities.cve	Recommended: The Common Vulnerabilities and Exposures .	object
vulnerabilities.cve.created_time	Recommended: The Record Creation Date identifies when the CVE ID was issued to a CVE Numbering Authority (CNA) or the CVE Record was published on the CVE List.	long
vulnerabilities.cve.cvss	Recommended: The CVSS object details Common Vulnerability Scoring System	array of objects
vulnerabilities.cve.cvss.base_score	Required: The CVSS base score. For example: `9.1`.	float
vulnerabilities.cve.cvss.depth	Recommended: The CVSS depth represents a depth of the equation used to calculate CVSS score.	string
vulnerabilities.cve.cvss.overall_score	Recommended: The CVSS overall score, impacted by base, temporal, and environmental metrics. For example: `9.1`.	float

Field	Description	Value Type
vulnerabilities.cve.cvss.version	Required: The CVSS version. For example: `3.1`.	string
vulnerabilities.cve.references	Recommended: A list of reference URLs with additional information about the CVE Record.	array of strings
vulnerabilities.cve.title	Recommended: A title or a brief phrase summarizing the CVE record.	string
vulnerabilities.cve.type	Recommended: <p>The vulnerability type as selected from a large dropdown menu during CVE refinement.	string
vulnerabilities.cve.uid	Required: The Common Vulnerabilities and Exposures unique number assigned to a specific computer vulnerability.	string
vulnerabilities.cwe	Recommended: The CWE object represents a weakness in a software system that can be exploited by a threat actor to perform an attack.	object
vulnerabilities.cwe.uid	Required: The Common Weakness Enumeration unique number assigned to a specific weakness.	string
vulnerabilities.references	Recommended: A list of reference URLs with additional information about the vulnerability.	array of strings

OCSF Identity & Access Management Schema

The Identity & Access Management class captures authentication, authorization, and identity-related events. This includes user login or logout events, authentication successes and failures, privilege escalations, group membership changes, and account management operations. This schema is part of the OCSF (Open

Cybersecurity Schema Framework) standard. For general OCSF guidance and field naming conventions, see "OCSF Schema Overview" on page 39.

For more information on the OCSF Version 1.2 standard, see OCSF v1.2 Documentation.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Field	Description	Value Type
action	Optional: The normalized caption of 'action_id' or the source-specific action.	string
action_id	Required: The normalized action taken by a control or other policy-based system leading to an outcome or disposition.	integer
activity_id	Required: The normalized identifier of the activity that triggered the event.	integer
activity_name	Optional: The event activity name, as defined by the activity_id.	string
actor	Optional: The actor object describes details about the user/role/process that was the source of the activity.	object
actor.app_uid	Optional: The unique identifier of the client application or service that initiated the activity.	string
actor.idp	Optional: This object describes details about the Identity Provider used.	object
actor.idp.name	Recommended: The name of the identity provider.	string
actor.idp.uid	Recommended: The unique identifier of the identity provider.	string

Field	Description	Value Type
actor.invoked_by	Optional: The name of the service that invoked the activity as described in the event.	string
actor.process	Recommended: The process that initiated the activity.	object
actor.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job. For example: `ssh user@10.0.0.10`. If the command line is unavailable or missing, the empty string `''` is to be used.	string
actor.process.container	Recommended: The information describing an instance of a container.	object
actor.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
actor.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string

Field	Description	Value Type
actor.process.container.name	Recommended: The container name.	string
actor.process.container.size	Recommended: The size of the container image.	long
actor.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`	string
actor.process.created_time	Recommended: The time when the process was created/started.	long
actor.process.file	Recommended: The process file object.	object
actor.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
actor.process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.file.name	Required: The name of the file. For example: `svchost.exe`	string
actor.process.file.parent_folder	Optional: The parent folder in which the file resides. For example: `"c:\windows\system32"`	string
actor.process.file.path	Recommended: The full path to the file. For example: `"c:\windows\system32\svchost.exe"`.	string
actor.process.file.size	Optional: The size of data, in bytes.	long
actor.process.file.type	Optional: The file type.	string
actor.process.file.type_id	Required: The file type ID.	integer
actor.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
actor.process.parent_process	Recommended: The parent process of this process object. It is recommended to only populate this field for the first process object, to prevent deep nesting.	object
actor.process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job. For example: `ssh user@10.0.0.10`. If the command line is unavailable or missing, the empty string `''` is to be used.	string
actor.process.parent_process.container	Recommended: The information describing an instance of a container.	object
actor.process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
actor.process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.container.name	Recommended: The container name.	string

Field	Description	Value Type
actor.process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`	string
actor.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
actor.process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string
actor.process.parent_process.file.path	Recommended: The full path to the file. For example: `"c:\windows\system32\svchost.exe"`.	string
actor.process.parent_process.file.type	Optional: The file type.	string
actor.process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process	Recommended: The parent process of this process object. It is recommended to only populate this field for the first process object, to prevent deep nesting.	object

Field	Description	Value Type
actor.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job. For example: `ssh user@10.0.0.10`. If the command line is unavailable or missing, the empty string `''` is to be used.	string
actor.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object
actor.process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
actor.process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string

Field	Description	Value Type
actor.process.parent_process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`	string
actor.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects

Field	Description	Value Type
actor.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string
actor.process.parent_process.parent_process.file.path	Recommended: The full path to the file. For example: `"c:\windows\system32\svchost.exe"`.	string
actor.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system. Process ID (PID) is a number used by the operating system to uniquely identify an active process.	integer

Field	Description	Value Type
actor.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool). Facilitates correlation of a process event with other events for that process.	string
actor.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
actor.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system. Process ID (PID) is a number used by the operating system to uniquely identify an active process.	integer
actor.process.parent_process.session	Optional: The user session under which this process is running.	object
actor.process.parent_process.session.created_time	Recommended: The time when the session was created.	long
actor.process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string

Field	Description	Value Type
actor.process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool). Facilitates correlation of a process event with other events for that process.	string
actor.process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
actor.process.pid	Recommended: The process identifier, as reported by the operating system. Process ID (PID) is a number used by the operating system to uniquely identify an active process.	integer
actor.process.session	Optional: The user session under which this process is running.	object
actor.process.session.created_time	Recommended: The time when the session was created.	long
actor.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean

Field	Description	Value Type
actor.process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.session.uuid	Optional: The universally unique identifier of the session.	string
actor.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool). Facilitates correlation of a process event with other events for that process.	string
actor.process.user	Recommended: The user under which this process is running.	object
actor.process.user.account	Optional: The user's account or the account associated with the user.	object
actor.process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.process.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.user.email_addr	Optional: The user's primary email address.	string

Field	Description	Value Type
actor.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.user.org	Optional: Organization and org unit related to the user.	object
actor.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.process.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
actor.process.user.type_id	Recommended: The account type identifier.	integer
actor.process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
actor.session	Optional: The user session from which the activity was initiated.	object
actor.session.created_time	Recommended: The time when the session was created.	long

Field	Description	Value Type
actor.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.session.issuer	Recommended: The identifier of the session issuer.	string
actor.session.uid	Recommended: The unique identifier of the session.	string
actor.session.uuid	Optional: The universally unique identifier of the session.	string
actor.user	Recommended: The user that initiated the activity or the user context from which the activity was initiated.	object
actor.user.account	Optional: The user's account or the account associated with the user.	object
actor.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.user.email_addr	Optional: The user's primary email address.	string

Field	Description	Value Type
actor.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.user.org	Optional: Organization and org unit related to the user.	object
actor.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
actor.user.type_id	Recommended: The account type identifier.	integer
actor.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
api	Optional: Describes details about a typical API (Application Programming Interface) call.	object

Field	Description	Value Type
api.operation	Required: Verb/Operation associated with the request	string
api.request	Recommended: Details pertaining to the API request.	object
api.request.uid	Required: The unique request identifier.	string
api.response	Recommended: Details pertaining to the API response.	object
api.response.code	Recommended: The numeric response sent to a request.	integer
api.response.error	Recommended: Error Code	string
api.response.error_message	Recommended: Error Message	string
api.response.message	Recommended: The description of the event/finding, as defined by the source.	string
api.service	Optional: The information pertaining to the API service.	object
api.service.name	Recommended: The name of the service.	string
api.service.uid	Recommended: The unique identifier of the service.	string
api.service.version	Recommended: The version of the service.	string
auth_protocol	Recommended: The authentication protocol as defined by the caption of 'auth_protocol_id'.	string
auth_protocol_id	Recommended: The normalized identifier of the authentication protocol used to create the user session.	integer

Field	Description	Value Type
category_name	Optional: The event category name, as defined by category_uid value.	string
category_uid	Required: The category unique identifier of the event.	integer
certificate	Recommended: The certificate associated with the authentication or pre-authentication (Kerberos).	object
certificate.created_time	Recommended: The time when the certificate was created.	long
certificate.expiration_time	Recommended: The expiration time of the certificate.	long
certificate.fingerprints	Required: The fingerprint list of the certificate.	array of objects
certificate.fingerprints.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
certificate.fingerprints.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
certificate.fingerprints.value	Required: The digital fingerprint value.	string
certificate.issuer	Required: The certificate issuer distinguished name.	string
certificate.serial_number	Required: The serial number of the certificate used to create the digital signature.	string
certificate.subject	Recommended: The certificate subject distinguished name.	string

Field	Description	Value Type
certificate.version	Recommended: The certificate version.	string
class_name	Optional: The event class name, as defined by the class_uid.	string
class_uid	Required: The unique identifier of a class. A class describes the attributes available in an event.	integer
cloud	Required: Describes details about the Cloud environment where the event was originally created or logged.	object
cloud.account	Optional: The account object describes details about the account that was the source or target of the activity.	object
cloud.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
cloud.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
cloud.account.type_id	Recommended: The normalized account type identifier.	integer
cloud.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
cloud.org	Optional: Organization and org unit relevant to the event or object.	object
cloud.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
cloud.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string

Field	Description	Value Type
cloud.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
cloud.provider	Required: The unique name of the Cloud services provider, such as AWS, MS Azure, GCP, etc.	string
cloud.region	Recommended: The name of the cloud region, as defined by the cloud provider.	string
comment	Recommended: The user provided comment about why the entity was changed.	string
count	Optional: The number of times that events in the same logical group occurred during the event Start Time to End Time period.	integer
device	Recommended: An addressable device, computer system or host.	object
device.container	Recommended: The information describing an instance of a container.	object
device.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
device.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
device.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
device.container.hash.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
device.container.image	Recommended: The container image used as a template to run the container.	object
device.container.image.name	Recommended: The image name. For example: `elixir`.	string
device.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
device.container.name	Recommended: The container name.	string
device.container.size	Recommended: The size of the container image.	long
device.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`.	string
device.domain	Optional: The network domain where the device resides. For example: `work.example.com`.	string
device.groups	Optional: The group names to which the device belongs. For example: `["Windows Laptops", "Engineering"]`.	array of objects
device.groups.desc	Optional: The group description.	string
device.groups.domain	Optional: The domain where the group is defined. For example: the LDAP or Active Directory domain.	string
device.groups.name	Recommended: The group name.	string
device.groups.type	Optional: The type of the group or account.	string

Field	Description	Value Type
device.groups.uid	Recommended: The unique identifier of the group. For example, for Windows events this is the security identifier (SID) of the group.	string
device.hostname	Recommended: The device hostname.	string
device.instance_uid	Recommended: The unique identifier of a VM instance.	string
device.interface_name	Recommended: The name of the network interface (For example, eth2).	string
device.interface_uid	Recommended: The unique identifier of the network interface.	string
device.ip	Recommended: The device IP address, in either IPv4 or IPv6 format.	string
device.is_compliant	Optional: The event occurred on a compliant device.	boolean
device.is_managed	Optional: The event occurred on a managed device.	boolean
device.name	Recommended: The alternate device name, ordinarily as assigned by an administrator.	string
device.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
device.os	Optional: The endpoint operating system.	object
device.os.name	Required: The operating system name.	string
device.os.type	Optional: The type of the operating system.	string

Field	Description	Value Type
device.os.type_id	Required: The type identifier of the operating system.	integer
device.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
device.owner.account	Optional: The user's account or the account associated with the user.	object
device.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
device.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
device.owner.account.type_id	Recommended: The normalized account type identifier.	integer
device.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
device.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
device.owner.email_addr	Optional: The user's primary email address.	string
device.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
device.owner.name	Recommended: The username. For example, `janedoe1`.	string
device.owner.org	Optional: Organization and org unit related to the user.	object

Field	Description	Value Type
device.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
device.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
device.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
device.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
device.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
device.owner.type_id	Recommended: The account type identifier.	integer
device.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
device.region	Recommended: The region where the virtual machine is located. For example, an AWS Region.	string
device.type	Optional: The device type.	string
device.type_id	Required: The device type ID.	integer
device.uid	Recommended: The unique identifier of the device. For example the Windows TargetSID or AWS EC2 ARN.	string
dst_endpoint	Recommended: The endpoint to which the authentication was targeted.	object

Field	Description	Value Type
dst_endpoint.container	Recommended: The information describing an instance of a container.	object
dst_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
dst_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
dst_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
dst_endpoint.container.name	Recommended: The container name.	string
dst_endpoint.container.size	Recommended: The size of the container image.	long
dst_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`.	string
dst_endpoint.domain	Optional: The name of the domain.	string

Field	Description	Value Type
dst_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
dst_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
dst_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
dst_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.location	Optional: The geographical location of the endpoint.	object
dst_endpoint.location.city	Recommended: The name of the city.	string
dst_endpoint.location.continent	Recommended: The name of the continent.	string
dst_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
dst_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: `42.361145`.	float
dst_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: `-71.057083`.	float
dst_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision of the country.	string
dst_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string

Field	Description	Value Type
dst_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
dst_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
dst_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
dst_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
dst_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
dst_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
dst_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
dst_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
dst_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
dst_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
dst_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
dst_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
dst_endpoint.owner.type_id	Recommended: The account type identifier.	integer
dst_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
dst_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
dst_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
dst_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
dst_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
dst_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
dst_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
dst_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
dst_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc...`.	string
dst_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string

Field	Description	Value Type
dst_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
dst_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
dst_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
dst_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
dst_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
dst_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
dst_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
dst_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string

Field	Description	Value Type
dst_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
dst_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
dst_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
dst_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
dst_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
dst_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
dst_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
dst_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
dst_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
dst_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
dst_endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.zone	Optional: The network zone or LAN segment.	string
duration	Optional: The event duration or aggregate time, the amount of time the event covers from `start_time` to `end_time` in milliseconds.	integer
end_time	Optional: The end time of a time period, or the time of the most recent event included in the aggregate event.	long
entity	Required: The managed entity that is being acted upon.	object

Field	Description	Value Type
entity.data	Optional: The managed entity content as a JSON object.	string
entity.name	Recommended: The name of the managed entity.	string
entity.type	Recommended: The managed entity type. For example: `policy`, `user`, `organizational unit`, `device`.	string
entity.uid	Recommended: The identifier of the managed entity.	string
entity.version	Recommended: The version of the managed entity. For example: `1.2.3`.	string
entity_result	Recommended: The updated managed entity.	object
entity_result.data	Optional: The managed entity content as a JSON object.	string
entity_result.name	Recommended: The name of the managed entity.	string
entity_result.type	Recommended: The managed entity type. For example: `policy`, `user`, `organizational unit`, `device`.	string
entity_result.uid	Recommended: The identifier of the managed entity.	string
entity_result.version	Recommended: The version of the managed entity. For example: `1.2.3`.	string
event_seq_id	Required: The sequence identifier for the event.	long
event_size	Required: The size of the event in bytes.	long

Field	Description	Value Type
event_time	Required: The normalized event occurrence time or the finding creation time.	long
group	Required: Group that was the target of the event.	object
group.desc	Optional: The group description.	string
group.domain	Optional: The domain where the group is defined. For example: the LDAP or Active Directory domain.	string
group.name	Recommended: The group name.	string
group.type	Optional: The type of the group or account.	string
group.uid	Recommended: The unique identifier of the group. For example, for Windows events this is the security identifier (SID) of the group.	string
http_request	Optional: Details about the underlying HTTP request.	object
http_request.http_headers	Recommended: Additional HTTP headers of an HTTP request or response.	array of objects
http_request.http_headers.name	Required: The name of the header	string
http_request.http_headers.value	Required: The value of the header	string

Field	Description	Value Type
http_request.http_method	Optional: The HTTP request method indicates the desired action to be performed for a given resource.	string
http_request.length	Optional: The HTTP request length, in number of bytes.	integer
http_request.referrer	Optional: The request header that identifies the address of the previous web page, which is linked to the current web page or resource being requested.	string
http_request.uid	Optional: The unique identifier of the http request.	string
http_request.url	Recommended: The URL object that pertains to the request.	object
http_request.url.category_ids	Recommended: The Website categorization identifiers.	array of integers
http_request.url.hostname	Recommended: The URL host as extracted from the URL. For example: `www.example.com` from `www.example.com/download/trouble`.	string
http_request.url.path	Recommended: The URL path as extracted from the URL. For example: `/download/trouble` from `www.example.com/download/trouble`.	string
http_request.url.port	Recommended: The URL port. For example: `80`.	integer

Field	Description	Value Type
http_request.url.query_string	Recommended: The query portion of the URL. For example: the query portion of the URL `http://www.example.com/search?q=bad&sort=date` is `q=bad&sort=date`.	string
http_request.url.scheme	Recommended: The scheme portion of the URL. For example: `http`, `https`, `ftp`, or `sftp`.	string
http_request.url.url_string	Recommended: The URL string. See RFC 1738. For example: `http://www.example.com/download/trouble.exe`. Note: The URL path should not populate the URL string.	string
http_request.user_agent	Recommended: The request header that identifies the operating system and web browser.	string
http_request.version	Recommended: The Hypertext Transfer Protocol (HTTP) version.	string
indexed_time	Required: The time when the event was indexed.	long
is_mfa	Recommended: Indicates whether Multi Factor Authentication was used during authentication.	boolean
is_remote	Recommended: The attempted authentication is over a remote connection.	boolean
logon_process	Optional: The trusted process that validated the authentication credentials.	object

Field	Description	Value Type
logon_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job. For example: `ssh user@10.0.0.10`. If the command line is unavailable or missing, the empty string `''` is to be used.	string
logon_process.container	Recommended: The information describing an instance of a container.	object
logon_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
logon_process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
logon_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
logon_process.container.hash.value	Required: The digital fingerprint value.	string
logon_process.container.image	Recommended: The container image used as a template to run the container.	object
logon_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
logon_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
logon_process.container.name	Recommended: The container name.	string

Field	Description	Value Type
logon_process.container.size	Recommended: The size of the container image.	long
logon_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`	string
logon_process.created_time	Recommended: The time when the process was created/started.	long
logon_process.file	Recommended: The process file object.	object
logon_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
logon_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
logon_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
logon_process.file.hashes	Recommended: An array of hash attributes.	array of objects
logon_process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
logon_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
logon_process.file.hashes.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
logon_process.file.name	Required: The name of the file. For example: `svchost.exe`	string
logon_process.file.parent_folder	Optional: The parent folder in which the file resides. For example: `"c:\windows\system32"`	string
logon_process.file.path	Recommended: The full path to the file. For example: `"c:\windows\system32\svchost.exe"`.	string
logon_process.file.size	Optional: The size of data, in bytes.	long
logon_process.file.type	Optional: The file type.	string
logon_process.file.type_id	Required: The file type ID.	integer
logon_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
logon_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
logon_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
logon_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
logon_process.parent_process	Recommended: The parent process of this process object. It is recommended to only populate this field for the first process object, to prevent deep nesting.	object

Field	Description	Value Type
logon_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job. For example: `ssh user@10.0.0.10`. If the command line is unavailable or missing, the empty string `''` is to be used.	string
logon_process.parent_process.container	Recommended: The information describing an instance of a container.	object
logon_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
logon_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
logon_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
logon_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
logon_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
logon_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
logon_process.parent_process.container.name	Recommended: The container name.	string
logon_process.parent_process.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
logon_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`	string
logon_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
logon_process.parent_process.file	Recommended: The process file object.	object
logon_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
logon_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
logon_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
logon_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
logon_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
logon_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
logon_process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string

Field	Description	Value Type
logon_process.parent_process.file.path	Recommended: The full path to the file. For example: `"c:\windows\system32\svchost.exe"`.	string
logon_process.parent_process.file.type	Optional: The file type.	string
logon_process.parent_process.file.type_id	Required: The file type ID.	integer
logon_process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
logon_process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
logon_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
logon_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
logon_process.parent_process.parent_process	Recommended: The parent process of this process object. It is recommended to only populate this field for the first process object, to prevent deep nesting.	object
logon_process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job. For example: `ssh user@10.0.0.10`. If the command line is unavailable or missing, the empty string `''` is to be used.	string

Field	Description	Value Type
logon_process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object
logon_process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
logon_process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
logon_process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
logon_process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
logon_process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
logon_process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
logon_process.parent_process.parent_process.container.name	Recommended: The container name.	string
logon_process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
logon_process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc....`	string
logon_process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
logon_process.parent_process.parent_process.file	Recommended: The process file object.	object
logon_process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
logon_process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
logon_process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
logon_process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
logon_process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
logon_process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
logon_process.parent_process.parent_process.file.name	Required: The name of the file. For example: `svchost.exe`	string
logon_process.parent_process.parent_process.file.path	Recommended: The full path to the file. For example: `"c:\windows\system32\svchost.exe"`.	string
logon_process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
logon_process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
logon_process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
logon_process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system. Process ID (PID) is a number used by the operating system to uniquely identify an active process.	integer
logon_process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool). Facilitates correlation of a process event with other events for that process.	string
logon_process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object

Field	Description	Value Type
logon_process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
logon_process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
logon_process.parent_process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
logon_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system. Process ID (PID) is a number used by the operating system to uniquely identify an active process.	integer
logon_process.parent_process.session	Optional: The user session under which this process is running.	object
logon_process.parent_process.session.created_time	Recommended: The time when the session was created.	long
logon_process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
logon_process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string
logon_process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
logon_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool). Facilitates correlation of a process event with other events for that process.	string

Field	Description	Value Type
logon_process.parent_process.user	Recommended: The user under which this process is running.	object
logon_process.parent_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
logon_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
logon_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
logon_process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
logon_process.pid	Recommended: The process identifier, as reported by the operating system. Process ID (PID) is a number used by the operating system to uniquely identify an active process.	integer
logon_process.session	Optional: The user session under which this process is running.	object
logon_process.session.created_time	Recommended: The time when the session was created.	long
logon_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
logon_process.session.issuer	Recommended: The identifier of the session issuer.	string
logon_process.session.uid	Recommended: The unique identifier of the session.	string

Field	Description	Value Type
logon_process.session.uuid	Optional: The universally unique identifier of the session.	string
logon_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool). Facilitates correlation of a process event with other events for that process.	string
logon_process.user	Recommended: The user under which this process is running.	object
logon_process.user.account	Optional: The user's account or the account associated with the user.	object
logon_ process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
logon_ process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
logon_ process.user.account.type_ id	Recommended: The normalized account type identifier.	integer
logon_ process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
logon_process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
logon_process.user.email_ addr	Optional: The user's primary email address.	string
logon_process.user.full_ name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
logon_process.user.name	Recommended: The username. For example, `janedoe1`.	string
logon_process.user.org	Optional: Organization and org unit related to the user.	object
logon_process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
logon_process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
logon_process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
logon_process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
logon_process.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
logon_process.user.type_id	Recommended: The account type identifier.	integer
logon_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
logon_type	Recommended: The logon type, normalized to the caption of the logon_type_id value.	string
logon_type_id	Recommended: The normalized logon type identifier.	integer
message	Recommended: The description of the event/finding, as defined by the source.	string

Field	Description	Value Type
metadata	Required: The metadata associated with the event or a finding.	object
metadata.correlation_uid	Optional: The unique identifier used to correlate events.	string
metadata.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.event_code	Optional: The Event ID or Code that the product uses to describe the event.	string
metadata.log_level	Optional: The audit level at which an event was generated.	string
metadata.log_name	Recommended: The event log name. For example, syslog file name or Windows logging subsystem: Security.	string
metadata.log_provider	Recommended: The logging provider or logging service that logged the event. For example, Microsoft-Windows-Security-Auditing.	string

Field	Description	Value Type
metadata.original_time	Recommended: The original event time as reported by the event source. For example, the time in the original format from system event log such as Syslog on Unix/Linux and the System event file on Windows. Omit if event is generated instead of collected via logs.	string
metadata.product	Required: The product that reported the event.	object
metadata.product.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.product.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.product.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.product.feature	Optional: The feature that reported the event.	object
metadata.product.feature.name	Recommended: The name of the feature.	string
metadata.product.feature.uid	Recommended: The unique identifier of the feature.	string
metadata.product.feature.version	Recommended: The version of the feature.	string
metadata.product.name	Recommended: The name of the product.	string
metadata.product.uid	Recommended: The unique identifier of the product.	string

Field	Description	Value Type
metadata.product.vendor_name	Required: The name of the vendor of the product.	string
metadata.product.version	Recommended: The version of the product, as defined by the event source. For example: `2013.1.3-beta`.	string
metadata.profiles	Optional: The list of profiles used to create the event.	array of strings
metadata.sequence	Optional: Sequence number of the event. The sequence number is a value available in some events, to make the exact ordering of events unambiguous, regardless of the event time precision.	integer
metadata.tenant_uid	Recommended: The unique tenant identifier.	string
metadata.uid	Optional: The logging system-assigned unique identifier of an event instance.	string
metadata.version	Required: The version of the OCSF schema, using Semantic Versioning Specification (SemVer). For example: 1.0.0. Event consumers use the version to determine the available event attributes.	string
observables	Recommended: The observables associated with the event or a finding.	array of objects

Field	Description	Value Type
observables.name	Required: The full name of the observable attribute. The `name` is a pointer/reference to an attribute within the event data. For example: `file.name`.	string
observables.type	Optional: The observable value type name.	string
observables.type_id	Required: The observable value type identifier.	integer
observables.value	Optional: The value associated with the observable attribute. The meaning of the value depends on the observable type.	string
original_event_time	Required: The original event time from the source system.	long
privileges	Required: List of privileges assigned to a user.	array of strings
raw_data	Optional: The raw event/finding data as received from the source.	string
raw_data_size	Optional: The size of the raw data which was transformed into an OCSF event, in bytes.	long
resource	Recommended: Resource that the privileges give access to.	object
resource.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
resource.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
resource.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
resource.owner	Recommended: The identity of the service or user account that owns the resource.	object
resource.owner.account	Optional: The user's account or the account associated with the user.	object
resource.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
resource.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
resource.owner.account.type_id	Recommended: The normalized account type identifier.	integer
resource.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
resource.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
resource.owner.email_addr	Optional: The user's primary email address.	string
resource.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
resource.owner.name	Recommended: The username. For example, `janedoe1`.	string
resource.owner.org	Optional: Organization and org unit related to the user.	object
resource.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
resource.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
resource.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
resource.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
resource.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
resource.owner.type_id	Recommended: The account type identifier.	integer
resource.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
service	Recommended: The service or gateway to which the user or process is being authenticated	object
service.name	Recommended: The name of the service.	string
service.uid	Recommended: The unique identifier of the service.	string

Field	Description	Value Type
service.version	Recommended: The version of the service.	string
session	Recommended: The authenticated user or service session.	object
session.created_time	Recommended: The time when the session was created.	long
session.is_remote	Recommended: The indication of whether the session is remote.	boolean
session.issuer	Recommended: The identifier of the session issuer.	string
session.uid	Recommended: The unique identifier of the session.	string
session.uuid	Optional: The universally unique identifier of the session.	string
severity	Optional: The event severity, normalized to the caption of the severity_id value.	string
severity_id	Required: The normalized identifier of the event/finding severity. The normalized severity is a measurement the effort and expense required to manage and resolve an event or incident. Smaller numerical values represent lower impact events, and larger numerical values represent higher impact events.	integer
src_endpoint	Recommended: Details about the source of the IAM activity.	object
src_endpoint.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
src_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
src_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
src_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
src_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
src_endpoint.container.name	Recommended: The container name.	string
src_endpoint.container.size	Recommended: The size of the container image.	long
src_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc...`.	string
src_endpoint.domain	Optional: The name of the domain.	string
src_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string

Field	Description	Value Type
src_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
src_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.location	Optional: The geographical location of the endpoint.	object
src_endpoint.location.city	Recommended: The name of the city.	string
src_endpoint.location.continent	Recommended: The name of the continent.	string
src_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
src_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: `42.361145`.	float
src_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: `-71.057083`.	float
src_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (For example, province or state) of the country.	string
src_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string

Field	Description	Value Type
src_endpoint.name	Recommended: The short name of the endpoint.	string
src_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
src_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
src_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
src_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
src_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.owner.email_addr	Optional: The user's primary email address.	string
src_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
src_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
src_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
src_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
src_endpoint.owner.type_id	Recommended: The account type identifier.	integer
src_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
src_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
src_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
src_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: `13550340a8681c84c861a...`.	object
src_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
src_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: `elixir`.	string
src_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
src_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
src_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
src_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: `ac2ea168264a08f9aaca0dfc...`.	string
src_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string

Field	Description	Value Type
src_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
src_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
src_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
src_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
src_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
src_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string

Field	Description	Value Type
src_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
src_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
src_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, `janedoe1`.	string
src_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
src_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
src_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
src_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
src_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.zone	Optional: The network zone or LAN segment.	string
start_time	Optional: The start time of a time period, or the time of the least recent event included in the aggregate event.	long
status	Recommended: The event status, normalized to the caption of the status_id value.	string
status_code	Recommended: The event status code, as reported by the event source.	string

Field	Description	Value Type
status_detail	Recommended: The status details contains additional information about the event/finding outcome.	string
status_id	Recommended: The normalized identifier of the event status.	integer
timezone_offset	Recommended: The number of minutes that the reported event `time` is ahead or behind UTC.	integer
type_name	Optional: The event type name, as defined by the type_uid.	string
type_uid	Required: The event/finding type ID. It identifies the event's semantics and structure.	long
unmapped	Optional: The attributes that are not mapped to the event schema. The names and values of those attributes are specific to the event source.	object
user	Required: User to which privileges were assigned.	object
user.account	Optional: The user's account or the account associated with the user.	object
user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
user.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
user.email_addr	Optional: The user's primary email address.	string
user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
user.name	Recommended: The username. For example, `janedoe1`.	string
user.org	Optional: Organization and org unit related to the user.	object
user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
user.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
user_result	Recommended: The result of the user account change. It should contain the new values of the changed attributes.	object
user_result.account	Optional: The user's account or the account associated with the user.	object
user_result.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
user_result.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
user_result.account.type_id	Recommended: The normalized account type identifier.	integer
user_result.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
user_result.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
user_result.email_addr	Optional: The user's primary email address.	string
user_result.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
user_result.name	Recommended: The username. For example, `janedoe1`.	string

Field	Description	Value Type
user_result.org	Optional: Organization and org unit related to the user.	object
user_result.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
user_result.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
user_result.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
user_result.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
user_result.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
user_result.type_id	Recommended: The account type identifier.	integer
user_result.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
volume	Required: The allocated event volume for the event in bytes.	long

OCSF Network Activity Schema

The Network Activity class captures network connection and traffic events across all network layers. This includes TCP/UDP connections, DNS queries, HTTP/HTTPS requests, network flow data, firewall events, and other protocol-level network activities. This schema is part of the OCSF (Open Cybersecurity Schema Framework) standard. For general OCSF guidance and field naming conventions, see "OCSF Schema Overview" on page 39.

For more information on the OCSF Version 1.2 standard, see OCSF v1.2 Documentation.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Field	Description	Value Type
action	Optional: The normalized caption of 'action_id' or the source-specific action.	string
action_id	Required: The normalized action taken by a control or other policy-based system leading to an outcome or disposition.	integer
activity_id	Required: The normalized identifier of the activity that triggered the event.	integer
activity_name	Optional: The event activity name, as defined by the activity_id.	string
actor	Optional: The actor object describes details about the user/role/process that was the source of the activity.	object
actor.app_uid	Optional: The unique identifier of the client application or service that initiated the activity.	string
actor.idp	Optional: This object describes details about the Identity Provider used.	object
actor.idp.name	Recommended: The name of the identity provider.	string

Field	Description	Value Type
actor.idp.uid	Recommended: The unique identifier of the identity provider.	string
actor.invoked_by	Optional: The name of the service that invoked the activity as described in the event.	string
actor.process	Recommended: The process that initiated the activity.	object
actor.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.container	Recommended: The information describing an instance of a container.	object
actor.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
actor.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
actor.process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.container.image.name	Recommended: The image name. For example: elixir.	string
actor.process.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
actor.process.container.name	Recommended: The container name.	string
actor.process.container.size	Recommended: The size of the container image.	long
actor.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
actor.process.created_time	Recommended: The time when the process was created/started.	long
actor.process.file	Recommended: The process file object.	object
actor.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
actor.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.file.name	Required: The name of the file. For example: svchost.exe	string
actor.process.file.parent_folder	Optional: The parent folder in which the file resides.	string
actor.process.file.path	Recommended: The full path to the file. .	string
actor.process.file.size	Optional: The size of data, in bytes.	long
actor.process.file.type	Optional: The file type.	string
actor.process.file.type_id	Required: The file type ID.	integer

Field	Description	Value Type
actor.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_ id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.name	Recommended: The friendly name of the process, for example: Notepad++.	string
actor.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.cmd_ line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_ process.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
actor.process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
actor.process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.container.image.name	Recommended: The image name. For example: elixir.	string
actor.process.parent_process.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
actor.process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string

Field	Description	Value Type
actor.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.file.name	Required: The name of the file. For example: svchost.exe	string
actor.process.parent_process.file.path	Recommended: The full path to the file. .	string
actor.process.parent_process.file.type	Optional: The file type.	string

Field	Description	Value Type
actor.process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.parent_process.name	Recommended: The friendly name of the process, for example: Notepad++.	string
actor.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
actor.process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
actor.process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: elixir.	string
actor.process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
actor.process.parent_process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string

Field	Description	Value Type
actor.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.file.name	Required: The name of the file. For example: svchost.exe	string
actor.process.parent_process.parent_process.file.path	Recommended: The full path to the file. .	string

Field	Description	Value Type
actor.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: Notepad++.	string
actor.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.parent_process.user.name	Recommended: The username. For example, janedoe1.	string
actor.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
actor.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.session	Optional: The user session under which this process is running.	object
actor.process.parent_ process.session.created_time	Recommended: The time when the session was created.	long
actor.process.parent_ process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.parent_ process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.parent_ process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_ process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.parent_ process.user.name	Recommended: The username. For example, janedoe1.	string

Field	Description	Value Type
actor.process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
actor.process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.session	Optional: The user session under which this process is running.	object
actor.process.session.created_time	Recommended: The time when the session was created.	long
actor.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.session.uuid	Optional: The universally unique identifier of the session.	string
actor.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string

Field	Description	Value Type
actor.process.user	Recommended: The user under which this process is running.	object
actor.process.user.account	Optional: The user's account or the account associated with the user.	object
actor.process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.process.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.process.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.process.user.email_addr	Optional: The user's primary email address.	string
actor.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.process.user.name	Recommended: The username. For example, janedoe1.	string

Field	Description	Value Type
actor.process.user.org	Optional: Organization and org unit related to the user.	object
actor.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.process.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string
actor.process.user.type_id	Recommended: The account type identifier.	integer
actor.process.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
actor.session	Optional: The user session from which the activity was initiated.	object

Field	Description	Value Type
actor.session.created_time	Recommended: The time when the session was created.	long
actor.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.session.issuer	Recommended: The identifier of the session issuer.	string
actor.session.uid	Recommended: The unique identifier of the session.	string
actor.session.uuid	Optional: The universally unique identifier of the session.	string
actor.user	Recommended: The user that initiated the activity or the user context from which the activity was initiated.	object
actor.user.account	Optional: The user's account or the account associated with the user.	object
actor.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.user.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
actor.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.user.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
actor.user.email_addr	Optional: The user's primary email address.	string
actor.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.user.name	Recommended: The username. For example, janedoe1.	string
actor.user.org	Optional: Organization and org unit related to the user.	object
actor.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string

Field	Description	Value Type
actor.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.user.type	Optional: The type of the user. For example, System, AWS IAM User,	string
actor.user.type_id	Recommended: The account type identifier.	integer
actor.user.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
answers	Recommended: The Domain Name System (DNS) answers.	array of objects
answers.class	Recommended: The class of DNS data contained in this resource record. See RFC1035. For example: IN.	string
answers.packet_uid	Recommended: The DNS packet identifier assigned by the program that generated the query.	integer
answers.rdata	Required: The data describing the DNS resource.	string
answers.ttl	Recommended: The time interval that the resource record may be cached.	integer

Field	Description	Value Type
answers.type	Recommended: The type of data contained in this resource record. See RFC1035. For example: CNAME.	string
api	Optional: Describes details about a typical API (Application Programming Interface) call.	object
api.operation	Required: Verb/Operation associated with the request	string
api.request	Recommended: Details pertaining to the API request.	object
api.request.uid	Required: The unique request identifier.	string
api.response	Recommended: Details pertaining to the API response.	object
api.response.code	Recommended: The numeric response sent to a request.	integer
api.response.error	Recommended: Error Code	string
api.response.error_message	Recommended: Error Message	string
api.response.message	Recommended: The description of the event/finding, as defined by the source.	string
api.service	Optional: The information pertaining to the API service.	object

Field	Description	Value Type
api.service.name	Recommended: The name of the service.	string
api.service.uid	Recommended: The unique identifier of the service.	string
api.service.version	Recommended: The version of the service.	string
app_name	Optional: The name of the application associated with the event or object.	string
auth_type	Recommended: The SSH authentication type, normalized to the caption of 'auth_type_id'.	string
auth_type_id	Recommended: The normalized identifier of the SSH authentication type.	integer
category_name	Optional: The event category name, as defined by category_uid value.	string
category_uid	Required: The category unique identifier of the event.	integer
certificate_chain	Recommended: The list of observed certificates in an RDP TLS connection.	array of strings
class_name	Optional: The event class name, as defined by the class_uid.	string
class_uid	Required: The unique identifier of a class. A class describes the attributes available in an event.	integer

Field	Description	Value Type
client_dialects	Recommended: The list of SMB dialects that the client speaks.	array of strings
client_hassh	Recommended: The Client HASSH fingerprinting object.	object
client_hassh.algorithm	Recommended: The concatenation of key exchange, encryption, authentication and compression algorithms.	string
client_hassh.fingerprint	Required: The hash of the key exchange, encryption, authentication and compression algorithms.	object
client_hassh.fingerprint.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
client_hassh.fingerprint.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
client_hassh.fingerprint.value	Required: The digital fingerprint value.	string
cloud	Required: Describes details about the Cloud environment where the event was originally created or logged.	object
cloud.account	Optional: The account object describes details about the account that was the source or target of the activity.	object

Field	Description	Value Type
cloud.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
cloud.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
cloud.account.type_id	Recommended: The normalized account type identifier.	integer
cloud.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
cloud.org	Optional: Organization and org unit relevant to the event or object.	object
cloud.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
cloud.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
cloud.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
cloud.provider	Required: The unique name of the Cloud services provider, such as AWS, MS Azure, GCP.	string

Field	Description	Value Type
cloud.region	Recommended: The name of the cloud region, as defined by the cloud provider.	string
codes	Recommended: The list of return codes to the FTP command.	array of integers
command	Recommended: The command name.	string
command_responses	Recommended: The list of responses to the FTP command.	array of strings
connection_info	Optional: The network connection information.	object
connection_info.direction	Optional: The direction of the initiated connection, traffic, or email, normalized to the caption of the direction_id value.	string
connection_info.direction_id	Required: The normalized identifier of the direction of the initiated connection, traffic, or email.	integer
connection_info.protocol_name	Optional: The TCP/IP protocol name in lowercase, as defined by the Internet Assigned Numbers Authority (IANA).	string
connection_info.protocol_num	Recommended: The TCP/IP protocol number, as defined by the Internet Assigned Numbers Authority (IANA).	integer

Field	Description	Value Type
connection_info.session	Optional: The authenticated user or service session.	object
connection_info.session.created_time	Recommended: The time when the session was created.	long
connection_info.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
connection_info.session.issuer	Recommended: The identifier of the session issuer.	string
connection_info.session.uid	Recommended: The unique identifier of the session.	string
connection_info.session.uuid	Optional: The universally unique identifier of the session.	string
connection_info.uid	Optional: The unique identifier of the connection.	string
count	Optional: The number of times that events in the same logical group occurred during the event Start Time to End Time period.	integer
delay	Recommended: The total round-trip delay to the reference clock in milliseconds.	integer
device	Recommended: An addressable device, computer system or host.	object

Field	Description	Value Type
device.container	Recommended: The information describing an instance of a container.	object
device.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
device.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
device.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
device.container.hash.value	Required: The digital fingerprint value.	string
device.container.image	Recommended: The container image used as a template to run the container.	object
device.container.image.name	Recommended: The image name. For example: elixir.	string
device.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
device.container.name	Recommended: The container name.	string

Field	Description	Value Type
device.container.size	Recommended: The size of the container image.	long
device.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
device.domain	Optional: The network domain where the device resides. For example: work.example.com.	string
device.groups	Optional: The group names to which the device belongs. For example: ["Windows Laptops", "Engineering"].	array of objects
device.groups.desc	Optional: The group description.	string
device.groups.domain	Optional: The domain where the group is defined. For example: the LDAP or Active Directory domain.	string
device.groups.name	Recommended: The group name.	string
device.groups.type	Optional: The type of the group or account.	string
device.groups.uid	Recommended: The unique identifier of the group.	string
device.hostname	Recommended: The device hostname.	string
device.instance_uid	Recommended: The unique identifier of a VM instance.	string

Field	Description	Value Type
device.interface_name	Recommended: The name of the network interface (For example, eth2).	string
device.interface_uid	Recommended: The unique identifier of the network interface.	string
device.ip	Recommended: The device IP address, in either IPv4 or IPv6 format.	string
device.is_compliant	Optional: The event occurred on a compliant device.	boolean
device.is_managed	Optional: The event occurred on a managed device.	boolean
device.name	Recommended: The alternate device name, ordinarily as assigned by an administrator.	string
device.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
device.os	Optional: The endpoint operating system.	object
device.os.name	Required: The operating system name.	string
device.os.type	Optional: The type of the operating system.	string
device.os.type_id	Required: The type identifier of the operating system.	integer

Field	Description	Value Type
device.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
device.owner.account	Optional: The user's account or the account associated with the user.	object
device.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
device.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
device.owner.account.type_id	Recommended: The normalized account type identifier.	integer
device.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
device.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
device.owner.email_addr	Optional: The user's primary email address.	string
device.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
device.owner.name	Recommended: The username. For example, janedoe1.	string
device.owner.org	Optional: Organization and org unit related to the user.	object
device.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
device.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
device.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
device.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
device.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
device.owner.type_id	Recommended: The account type identifier.	integer
device.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
device.region	Recommended: The region where the virtual machine is located. For example, an AWS Region.	string
device.type	Optional: The device type.	string
device.type_id	Required: The device type ID.	integer
device.uid	Recommended: The unique identifier of the device. For example the Windows TargetSID or AWS EC2 ARN.	string
dialect	Recommended: The negotiated protocol dialect.	string
direction	Optional: The direction of the email, as defined by the direction_id value.	string
direction_id	Required: The direction of the email relative to the scanning host or organization.	integer
dispersion	Recommended: The dispersion in the NTP protocol is the estimated time error or uncertainty relative to the reference clock in milliseconds.	integer
dst_endpoint	Recommended: The responder (server) of the DHCP connection.	object

Field	Description	Value Type
dst_endpoint.container	Recommended: The information describing an instance of a container.	object
dst_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
dst_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.container.image.name	Recommended: The image name. For example: elixir.	string
dst_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
dst_endpoint.container.name	Recommended: The container name.	string

Field	Description	Value Type
dst_endpoint.container.size	Recommended: The size of the container image.	long
dst_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
dst_endpoint.domain	Optional: The name of the domain.	string
dst_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
dst_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
dst_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
dst_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.location	Optional: The geographical location of the endpoint.	object
dst_endpoint.location.city	Recommended: The name of the city.	string
dst_endpoint.location.continent	Recommended: The name of the continent.	string

Field	Description	Value Type
dst_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
dst_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: 42.361145.	float
dst_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: -71.057083.	float
dst_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (For example, province or state) of the country.	string
dst_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
dst_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
dst_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object

Field	Description	Value Type
dst_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
dst_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
dst_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
dst_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
dst_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
dst_endpoint.owner.name	Recommended: The username. For example, janedoe1.	string
dst_endpoint.owner.org	Optional: Organization and org unit related to the user.	object

Field	Description	Value Type
dst_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
dst_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
dst_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
dst_endpoint.owner.type_id	Recommended: The account type identifier.	integer
dst_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
dst_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object

Field	Description	Value Type
dst_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object
dst_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
dst_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
dst_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
dst_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
dst_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
dst_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: elixir.	string
dst_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
dst_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string

Field	Description	Value Type
dst_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
dst_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
dst_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
dst_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
dst_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
dst_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
dst_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
dst_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
dst_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
dst_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
dst_endpoint.proxy_ endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
dst_endpoint.proxy_ endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
dst_endpoint.proxy_ endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
dst_endpoint.proxy_ endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
dst_endpoint.proxy_ endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
dst_endpoint.proxy_ endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
dst_endpoint.proxy_ endpoint.owner.email_addr	Optional: The user's primary email address.	string
dst_endpoint.proxy_ endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
dst_endpoint.proxy_ endpoint.owner.name	Recommended: The username. For example, janedoe1.	string
dst_endpoint.proxy_ endpoint.owner.org	Optional: Organization and org unit related to the user.	object
dst_endpoint.proxy_ endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
dst_endpoint.proxy_ endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
dst_endpoint.proxy_ endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
dst_endpoint.proxy_ endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
dst_endpoint.proxy_ endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
dst_endpoint.proxy_ endpoint.owner.type_id	Recommended: The account type identifier.	integer
dst_endpoint.proxy_ endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
dst_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
dst_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
dst_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
dst_endpoint.type_id	Recommended: The network endpoint type ID.	integer
dst_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
dst_endpoint.zone	Optional: The network zone or LAN segment.	string
duration	Optional: The event duration or aggregate time, the amount of time the event covers from start_time to end_time in milliseconds.	integer
email	Required: The email object.	object
email.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
email.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
email.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
email.from	Required: The email header From values, as defined by RFC 5322.	string
email.message_uid	Recommended: The email header Message-Id value, as defined by RFC 5322.	string
email.reply_to	Recommended: The email header Reply-To values, as defined by RFC 5322.	string
email.size	Recommended: The size in bytes of the email, including attachments.	long
email.smtp_from	Recommended: The value of the SMTP MAIL FROM command.	string
email.smtp_to	Recommended: The value of the SMTP envelope RCPT TO command.	array of strings
email.subject	Optional: The email header Subject value, as defined by RFC 5322.	string
email.to	Required: The email header To values, as defined by RFC 5322.	array of strings

Field	Description	Value Type
email.uid	Recommended: The email unique identifier.	string
email_auth	Recommended: The SPF, DKIM and DMARC attributes of an email.	object
email_auth.dkim	Recommended: The DomainKeys Identified Mail (DKIM) status of the email.	string
email_auth.dkim_domain	Recommended: The DomainKeys Identified Mail (DKIM) signing domain of the email.	string
email_auth.dkim_signature	Recommended: The DomainKeys Identified Mail (DKIM) signature used by the sending/receiving system.	string
email_auth.dmarc	Recommended: The Domain-based Message Authentication, Reporting and Conformance (DMARC) status of the email.	string
email_auth.dmarc_override	Recommended: The Domain-based Message Authentication, Reporting and Conformance (DMARC) override action.	string
email_auth.dmarc_policy	Recommended: The Domain-based Message Authentication, Reporting and Conformance (DMARC) policy status.	string

Field	Description	Value Type
email_auth.spf	Recommended: The Sender Policy Framework (SPF) status of the email.	string
email_uid	Required: The unique identifier of the email, used to correlate related email alert and activity events.	string
end_time	Optional: The end time of a time period, or the time of the most recent event included in the aggregate event.	long
event_seq_id	Required: The sequence identifier for the event.	long
event_size	Required: The size of the event in bytes.	long
event_time	Required: The normalized event occurrence time or the finding creation time.	long
file	Optional: The file that is the target of the SSH activity.	object
file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
file.hashes	Recommended: An array of hash attributes.	array of objects
file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
file.hashes.value	Required: The digital fingerprint value.	string
file.name	Required: The name of the file. For example: svchost.exe	string
file.parent_folder	Optional: The parent folder in which the file resides.	string
file.path	Recommended: The full path to the file. .	string
file.size	Optional: The size of data, in bytes.	long
file.type	Optional: The file type.	string
file.type_id	Required: The file type ID.	integer
http_cookies	Recommended: The cookies object describes details about HTTP cookies	array of objects

Field	Description	Value Type
http_cookies.name	Required: The HTTP cookie name.	string
http_cookies.value	Required: The HTTP cookie value.	string
http_request	Required: The HTTP Request Object documents attributes of a request made to a web server.	object
http_request.http_headers	Recommended: Additional HTTP headers of an HTTP request or response.	array of objects
http_request.http_headers.name	Required: The name of the header	string
http_request.http_headers.value	Required: The value of the header	string
http_request.http_method	Optional: The HTTP request method indicates the desired action to be performed for a given resource.	string
http_request.length	Optional: The HTTP request length, in number of bytes.	integer
http_request.referrer	Optional: The request header that identifies the address of the previous web page, which is linked to the current web page or resource being requested.	string
http_request.uid	Optional: The unique identifier of the http request.	string

Field	Description	Value Type
http_request.url	Recommended: The URL object that pertains to the request.	object
http_request.url.category_ids	Recommended: The Website categorization identifiers.	array of integers
http_request.url.hostname	Recommended: The URL host as extracted from the URL.	string
http_request.url.path	Recommended: The URL path as extracted from the URL.	string
http_request.url.port	Recommended: The URL port. For example: 80.	integer
http_request.url.query_string	Recommended: The query portion of the URL.	string
http_request.url.scheme	Recommended: The scheme portion of the URL. For example: http, https, ftp, or sftp.	string
http_request.url.url_string	Recommended: The URL string.	string
http_request.user_agent	Recommended: The request header that identifies the operating system and web browser.	string
http_request.version	Recommended: The Hypertext Transfer Protocol (HTTP) version.	string
http_response	Required: The HTTP Response from a web server to a requester.	object

Field	Description	Value Type
http_response.code	Required: The Hypertext Transfer Protocol (HTTP) status code returned from the web server to the client. For example, 200.	integer
http_response.content_type	Optional: The request header that identifies the original media type of the resource (prior to any content encoding applied for sending).	string
http_response.http_headers	Recommended: Additional HTTP headers of an HTTP request or response.	array of objects
http_response.http_headers.name	Required: The name of the header	string
http_response.http_headers.value	Required: The value of the header	string
indexed_time	Required: The time when the event was indexed.	long
is_renewal	Recommended: The indication of whether this is a lease/session renewal event.	boolean
lease_dur	Recommended: This represents the length of the DHCP lease in seconds.	integer
load_balancer	Recommended: The Load Balancer object contains information related to the device that is distributing incoming traffic to specified destinations.	object

Field	Description	Value Type
load_balancer.code	Recommended: The numeric response status code detailing the connection from the load balancer to the destination target.	integer
load_balancer.dst_endpoint	Recommended: The destination to which the load balancer is distributing traffic.	object
load_balancer.dst_endpoint.container	Recommended: The information describing an instance of a container.	object
load_balancer.dst_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
load_balancer.dst_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
load_balancer.dst_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
load_balancer.dst_endpoint.container.hash.value	Required: The digital fingerprint value.	string
load_balancer.dst_endpoint.container.image	Recommended: The container image used as a template to run the container.	object

Field	Description	Value Type
load_balancer.dst_endpoint.container.image.name	Recommended: The image name. For example: elixir.	string
load_balancer.dst_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
load_balancer.dst_endpoint.container.name	Recommended: The container name.	string
load_balancer.dst_endpoint.container.size	Recommended: The size of the container image.	long
load_balancer.dst_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
load_balancer.dst_endpoint.domain	Optional: The name of the domain.	string
load_balancer.dst_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
load_balancer.dst_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
load_balancer.dst_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
load_balancer.dst_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
load_balancer.dst_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string

Field	Description	Value Type
load_balancer.dst_endpoint.location	Optional: The geographical location of the endpoint.	object
load_balancer.dst_endpoint.location.city	Recommended: The name of the city.	string
load_balancer.dst_endpoint.location.continent	Recommended: The name of the continent.	string
load_balancer.dst_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
load_balancer.dst_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD)	float
load_balancer.dst_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD).	float
load_balancer.dst_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision of the country.	string
load_balancer.dst_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
load_balancer.dst_endpoint.name	Recommended: The short name of the endpoint.	string
load_balancer.dst_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
load_balancer.dst_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
load_balancer.dst_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
load_balancer.dst_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
load_balancer.dst_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
load_balancer.dst_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
load_balancer.dst_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
load_balancer.dst_endpoint.owner.domain	Optional: The domain where the user is defined.	string
load_balancer.dst_endpoint.owner.email_addr	Optional: The user's primary email address.	string
load_balancer.dst_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
load_balancer.dst_endpoint.owner.name	Recommended: The username. For example, janedoe1.	string

Field	Description	Value Type
load_balancer.dst_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
load_balancer.dst_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
load_balancer.dst_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
load_balancer.dst_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
load_balancer.dst_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
load_balancer.dst_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
load_balancer.dst_endpoint.owner.type_id	Recommended: The account type identifier.	integer
load_balancer.dst_endpoint.owner.uid	Recommended: The unique user identifier.	string
load_balancer.dst_endpoint.port	Recommended: The port used for communication within the network connection.	integer
load_balancer.dst_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object

Field	Description	Value Type
load_balancer.dst_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object
load_balancer.dst_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
load_balancer.dst_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
load_balancer.dst_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
load_balancer.dst_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string
load_balancer.dst_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
load_balancer.dst_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: elixir.	string
load_balancer.dst_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
load_balancer.dst_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string

Field	Description	Value Type
load_balancer.dst_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
load_balancer.dst_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
load_balancer.dst_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
load_balancer.dst_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
load_balancer.dst_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
load_balancer.dst_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
load_balancer.dst_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
load_balancer.dst_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
load_balancer.dst_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
load_balancer.dst_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
load_balancer.dst_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
load_balancer.dst_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
load_balancer.dst_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
load_balancer.dst_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
load_balancer.dst_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
load_balancer.dst_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
load_balancer.dst_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
load_balancer.dst_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
load_balancer.dst_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, janedoe1.	string
load_balancer.dst_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
load_balancer.dst_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
load_balancer.dst_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
load_balancer.dst_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
load_balancer.dst_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
load_balancer.dst_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
load_balancer.dst_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer
load_balancer.dst_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
load_balancer.dst_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
load_balancer.dst_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
load_balancer.dst_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
load_balancer.dst_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
load_balancer.dst_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
load_balancer.dst_endpoint.type_id	Recommended: The network endpoint type ID.	integer
load_balancer.dst_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
load_balancer.dst_endpoint.zone	Optional: The network zone or LAN segment.	string
load_balancer.endpoint_connections	Recommended: An object detailing the load balancer connection attempts and responses.	array of objects
load_balancer.endpoint_connections.code	Recommended: A numerical response status code providing details about the connection.	integer
load_balancer.endpoint_connections.network_endpoint	Recommended: Provides characteristics of the network endpoint.	object

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.container	Recommended: The information describing an instance of a container.	object
load_balancer.endpoint_connections.network_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
load_balancer.endpoint_connections.network_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
load_balancer.endpoint_connections.network_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
load_balancer.endpoint_connections.network_endpoint.container.hash.value	Required: The digital fingerprint value.	string
load_balancer.endpoint_connections.network_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
load_balancer.endpoint_connections.network_endpoint.container.image.name	Recommended: The image name. For example: elixir.	string
load_balancer.endpoint_connections.network_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.container.name	Recommended: The container name.	string
load_balancer.endpoint_connections.network_endpoint.container.size	Recommended: The size of the container image.	long
load_balancer.endpoint_connections.network_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
load_balancer.endpoint_connections.network_endpoint.domain	Optional: The name of the domain.	string
load_balancer.endpoint_connections.network_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
load_balancer.endpoint_connections.network_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
load_balancer.endpoint_connections.network_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
load_balancer.endpoint_connections.network_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
load_balancer.endpoint_connections.network_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.location	Optional: The geographical location of the endpoint.	object
load_balancer.endpoint_connections.network_endpoint.location.city	Recommended: The name of the city.	string
load_balancer.endpoint_connections.network_endpoint.location.continent	Recommended: The name of the continent.	string
load_balancer.endpoint_connections.network_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
load_balancer.endpoint_connections.network_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: 42.361145.	float
load_balancer.endpoint_connections.network_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: -71.057083.	float
load_balancer.endpoint_connections.network_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision of the country.	string
load_balancer.endpoint_connections.network_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
load_balancer.endpoint_connections.network_endpoint.name	Recommended: The short name of the endpoint.	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
load_balancer.endpoint_connections.network_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
load_balancer.endpoint_connections.network_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
load_balancer.endpoint_connections.network_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
load_balancer.endpoint_connections.network_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
load_balancer.endpoint_connections.network_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
load_balancer.endpoint_connections.network_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
load_balancer.endpoint_connections.network_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
load_balancer.endpoint_connections.network_endpoint.owner.email_addr	Optional: The user's primary email address.	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
load_balancer.endpoint_connections.network_endpoint.owner.name	Recommended: The username. For example, janedoe1.	string
load_balancer.endpoint_connections.network_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
load_balancer.endpoint_connections.network_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
load_balancer.endpoint_connections.network_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
load_balancer.endpoint_connections.network_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
load_balancer.endpoint_connections.network_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
load_balancer.endpoint_connections.network_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
load_balancer.endpoint_connections.network_endpoint.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
load_balancer.endpoint_connections.network_endpoint.port	Recommended: The port used for communication within the network connection.	integer
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container	Recommended: The information describing an instance of a container.	object
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.hash.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.image.name	Recommended: The image name. For example: elixir.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.email_addr	Optional: The user's primary email address.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.name	Recommended: The username. For example, janedoe1.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.type_id	Recommended: The account type identifier.	integer
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
load_balancer.endpoint_connections.network_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
load_balancer.endpoint_connections.network_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string

Field	Description	Value Type
load_balancer.endpoint_connections.network_endpoint.type_id	Recommended: The network endpoint type ID.	integer
load_balancer.endpoint_connections.network_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
load_balancer.endpoint_connections.network_endpoint.zone	Optional: The network zone or LAN segment.	string
load_balancer.name	Recommended: The name of the load balancer.	string
load_balancer.uid	Recommended: The unique identifier for the load balancer.	string
message	Recommended: The description of the event/finding, as defined by the source.	string
metadata	Required: The metadata associated with the event or a finding.	object
metadata.correlation_uid	Optional: The unique identifier used to correlate events.	string
metadata.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
metadata.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.event_code	Optional: The Event ID or Code that the product uses to describe the event.	string
metadata.log_level	Optional: The audit level at which an event was generated.	string
metadata.log_name	Recommended: The event log name. For example, syslog file name or Windows logging subsystem: Security.	string
metadata.log_provider	Recommended: The logging provider or logging service that logged the event.	string
metadata.original_time	Recommended: The original event time as reported by the event source.	string
metadata.product	Required: The product that reported the event.	object
metadata.product.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.product.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
metadata.product.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.product.feature	Optional: The feature that reported the event.	object
metadata.product.feature.name	Recommended: The name of the feature.	string
metadata.product.feature.uid	Recommended: The unique identifier of the feature.	string
metadata.product.feature.version	Recommended: The version of the feature.	string
metadata.product.name	Recommended: The name of the product.	string
metadata.product.uid	Recommended: The unique identifier of the product.	string
metadata.product.vendor_name	Required: The name of the vendor of the product.	string
metadata.product.version	Recommended: The version of the product, as defined by the event source. For example: 2013.1.3-beta.	string
metadata.profiles	Optional: The list of profiles used to create the event.	array of strings
metadata.sequence	Optional: Sequence number of the event.	integer
metadata.tenant_uid	Recommended: The unique tenant identifier.	string

Field	Description	Value Type
metadata.uid	Optional: The logging system-assigned unique identifier of an event instance.	string
metadata.version	Required: The version of the OCSF schema, using Semantic Versioning Specification (SemVer).	string
name	Recommended: The name of the data affiliated with the command.	string
observables	Recommended: The observables associated with the event or a finding.	array of objects
observables.name	Required: The full name of the observable attribute. The `name` is a pointer/reference to an attribute within the event data. For example: `file.name`.	string
observables.type	Optional: The observable value type name.	string
observables.type_id	Required: The observable value type identifier.	integer
observables.value	Optional: The value associated with the observable attribute. The meaning of the value depends on the observable type.	string

Field	Description	Value Type
open_type	Recommended: Indicates how the file was opened (For example, normal, delete on close).	string
original_event_time	Required: The original event time from the source system.	long
port	Recommended: The dynamic port established for impending data transfers.	integer
precision	Recommended: The NTP precision quantifies a clock's accuracy and stability in log2 seconds, as defined in RFC-5905.	integer
protocol_ver	Recommended: The Secure Shell Protocol version.	string
proxy	Recommended: The proxy (server) in a network connection.	object
proxy.container	Recommended: The information describing an instance of a container.	object
proxy.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object

Field	Description	Value Type
proxy.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
proxy.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
proxy.container.hash.value	Required: The digital fingerprint value.	string
proxy.container.image	Recommended: The container image used as a template to run the container.	object
proxy.container.image.name	Recommended: The image name. For example: elixir.	string
proxy.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
proxy.container.name	Recommended: The container name.	string
proxy.container.size	Recommended: The size of the container image.	long
proxy.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
proxy.hostname	Recommended: The fully qualified name of the endpoint.	string

Field	Description	Value Type
proxy.instance_uid	Recommended: The unique identifier of a VM instance.	string
proxy.interface_name	Recommended: The name of the network interface (For example, eth2).	string
proxy.interface_uid	Recommended: The unique identifier of the network interface.	string
proxy.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
proxy.name	Recommended: The short name of the endpoint.	string
proxy.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
proxy.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
proxy.owner.account	Optional: The user's account or the account associated with the user.	object
proxy.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string

Field	Description	Value Type
proxy.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
proxy.owner.account.type_id	Recommended: The normalized account type identifier.	integer
proxy.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
proxy.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
proxy.owner.email_addr	Optional: The user's primary email address.	string
proxy.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
proxy.owner.name	Recommended: The username. For example, janedoe1.	string
proxy.owner.org	Optional: Organization and org unit related to the user.	object
proxy.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
proxy.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string

Field	Description	Value Type
proxy.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
proxy.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
proxy.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
proxy.owner.type_id	Recommended: The account type identifier.	integer
proxy.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
proxy.port	Recommended: The port used for communication within the network connection.	integer
proxy.svc_name	Recommended: The service name in service-to-service connections.	string
proxy.type_id	Recommended: The network endpoint type ID.	integer
proxy.uid	Recommended: The unique identifier of the endpoint.	string

Field	Description	Value Type
proxy_connection_info	Recommended: The connection information from the proxy server to the remote server.	object
proxy_connection_info.direction	Optional: The direction of the initiated connection, traffic, or email, normalized to the caption of the direction_id value.	string
proxy_connection_info.direction_id	Required: The normalized identifier of the direction of the initiated connection, traffic, or email.	integer
proxy_connection_info.protocol_name	Optional: The TCP/IP protocol name in lowercase, as defined by the Internet Assigned Numbers Authority (IANA).	string
proxy_connection_info.protocol_num	Recommended: The TCP/IP protocol number, as defined by the Internet Assigned Numbers Authority (IANA).	integer
proxy_connection_info.session	Optional: The authenticated user or service session.	object
proxy_connection_ info.session.created_time	Recommended: The time when the session was created.	long
proxy_connection_info.session.is_ remote	Recommended: The indication of whether the session is remote.	boolean

Field	Description	Value Type
proxy_connection_info.session.issuer	Recommended: The identifier of the session issuer.	string
proxy_connection_info.session.uid	Recommended: The unique identifier of the session.	string
proxy_connection_info.session.uuid	Optional: The universally unique identifier of the session.	string
proxy_connection_info.uid	Optional: The unique identifier of the connection.	string
proxy_tls	Recommended: The TLS protocol negotiated between the proxy server and the remote server.	object
proxy_tls.certificate	Recommended: The certificate object containing information about the digital certificate.	object
proxy_tls.certificate.created_time	Recommended: The time when the certificate was created.	long
proxy_tls.certificate.expiration_time	Recommended: The expiration time of the certificate.	long
proxy_tls.certificate.fingerprints	Required: The fingerprint list of the certificate.	array of objects
proxy_ tls.certificate.fingerprints.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string

Field	Description	Value Type
proxy_ tls.certificate.fingerprints.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
proxy_tls.certificate.fingerprints.value	Required: The digital fingerprint value.	string
proxy_tls.certificate.issuer	Required: The certificate issuer distinguished name.	string
proxy_tls.certificate.serial_number	Required: The serial number of the certificate used to create the digital signature.	string
proxy_tls.certificate.subject	Recommended: The certificate subject distinguished name.	string
proxy_tls.certificate.version	Recommended: The certificate version.	string
proxy_tls.certificate_chain	Recommended: The Chain of Certificate Serial Numbers field provides a chain of Certificate Issuer Serial Numbers leading to the Root Certificate Issuer.	array of strings
proxy_tls.cipher	Recommended: The negotiated cipher suite.	string
proxy_tls.client_ciphers	Recommended: The client cipher suites that were exchanged during the TLS handshake negotiation.	array of strings
proxy_tls.ja3_hash	Recommended: The MD5 hash of a JA3 string.	object

Field	Description	Value Type
proxy_tls.ja3_hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
proxy_tls.ja3_hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
proxy_tls.ja3_hash.value	Required: The digital fingerprint value.	string
proxy_tls.ja3s_hash	Recommended: The MD5 hash of a JA3S string.	object
proxy_tls.ja3s_hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
proxy_tls.ja3s_hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
proxy_tls.ja3s_hash.value	Required: The digital fingerprint value.	string
proxy_tls.sni	Recommended: The Server Name Indication (SNI) extension sent by the client.	string
proxy_tls.version	Required: The TLS protocol version.	string

Field	Description	Value Type
proxy_traffic	Recommended: The network traffic refers to the amount of data moving across a network, from proxy to remote server at a given point of time.	object
proxy_traffic.bytes	Recommended: The total number of bytes (in and out).	long
proxy_traffic.bytes_in	Optional: The number of bytes sent from the destination to the source.	long
proxy_traffic.bytes_out	Optional: The number of bytes sent from the source to the destination.	long
proxy_traffic.chunks	Optional: The total number of chunks (in and out).	long
proxy_traffic.chunks_out	Optional: The number of chunks sent from the source to the destination.	long
proxy_traffic.packets	Recommended: The total number of packets (in and out).	long
proxy_traffic.packets_in	Optional: The number of packets sent from the destination to the source.	long
proxy_traffic.packets_out	Optional: The number of packets sent from the source to the destination.	long
query	Recommended: The Domain Name System (DNS) query.	object

Field	Description	Value Type
query.class	Recommended: The class of resource records being queried. See RFC1035. For example: IN.	string
query.hostname	Required: The hostname or domain being queried. For example: www.example.com	string
query.opcode_id	Recommended: The DNS opcode ID specifies the normalized query message type.	integer
query.packet_uid	Recommended: The DNS packet identifier assigned by the program that generated the query.	integer
query.type	Recommended: The type of resource records being queried.	string
query_time	Recommended: The Domain Name System (DNS) query time.	long
raw_data	Optional: The raw event/finding data as received from the source.	string
raw_data_size	Optional: The size of the raw data which was transformed into an OCSF event, in bytes.	long
rcode	Recommended: The DNS server response code, normalized to the caption of the rcode_id value.	string

Field	Description	Value Type
rcode_id	Recommended: The normalized identifier of the DNS server response code. See RFC-6895.	integer
relay	Recommended: The network relay that is associated with the event.	object
relay.hostname	Recommended: The hostname associated with the network interface.	string
relay.ip	Recommended: The IP address associated with the network interface.	string
relay.mac	Recommended: The MAC address of the network interface.	string
relay.name	Recommended: The name of the network interface.	string
relay.type_id	Required: The network interface type identifier.	integer
request	Recommended: The client request in an RDP network connection.	object
request.uid	Required: The unique request identifier.	string
response	Recommended: The server response in an SMB network connection.	object
response.code	Recommended: The numeric response sent to a request.	integer

Field	Description	Value Type
response.error	Recommended: Error Code	string
response.error_message	Recommended: Error Message	string
response.message	Recommended: The description of the event/finding, as defined by the source.	string
response_time	Recommended: The Domain Name System (DNS) response time.	long
server_hassh	Recommended: The Server HASSH fingerprinting object.	object
server_hassh.algorithm	Recommended: The concatenation of key exchange, encryption, authentication and compression algorithms.	string
server_hassh.fingerprint	Required: The hash of the key exchange, encryption, authentication and compression algorithms.	object
server_hassh.fingerprint.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
server_hassh.fingerprint.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
server_hassh.fingerprint.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
session	Recommended: The session associated with the tunnel.	object
session.created_time	Recommended: The time when the session was created.	long
session.is_remote	Recommended: The indication of whether the session is remote.	boolean
session.issuer	Recommended: The identifier of the session issuer.	string
session.uid	Recommended: The unique identifier of the session.	string
session.uuid	Optional: The universally unique identifier of the session.	string
severity	Optional: The event severity, normalized to the caption of the severity_id value.	string
severity_id	Required: The normalized identifier of the event/finding severity.	integer
share	Recommended: The SMB share name.	string
share_type	Recommended: The SMB share type, normalized to the caption of the share_type_id value.	string
share_type_id	Recommended: The normalized identifier of the SMB share type.	integer

Field	Description	Value Type
smtp_hello	Recommended: The value of the SMTP HELO or EHLO command sent by the initiator (client).	string
src_endpoint	Recommended: The initiator (client) of the DHCP connection.	object
src_endpoint.container	Recommended: The information describing an instance of a container.	object
src_endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861aa....	object
src_endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
src_endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.container.image.name	Recommended: The image name. For example: elixir.	string

Field	Description	Value Type
src_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
src_endpoint.container.name	Recommended: The container name.	string
src_endpoint.container.size	Recommended: The size of the container image.	long
src_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
src_endpoint.domain	Optional: The name of the domain.	string
src_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
src_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
src_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.location	Optional: The geographical location of the endpoint.	object

Field	Description	Value Type
src_endpoint.location.city	Recommended: The name of the city.	string
src_endpoint.location.continent	Recommended: The name of the continent.	string
src_endpoint.location.country	Recommended: The ISO 3166-1 Alpha-2 country code.	string
src_endpoint.location.lat	Optional: The geographical Latitude coordinate represented in Decimal Degrees (DD). For example: 42.361145.	float
src_endpoint.location.long	Optional: The geographical Longitude coordinate represented in Decimal Degrees (DD). For example: -71.057083.	float
src_endpoint.location.region	Optional: The alphanumeric code that identifies the principal subdivision (For example, province or state) of the country.	string
src_endpoint.mac	Optional: The Media Access Control (MAC) address of the endpoint.	string
src_endpoint.name	Recommended: The short name of the endpoint.	string
src_endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
src_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
src_endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
src_endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
src_endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.owner.email_addr	Optional: The user's primary email address.	string
src_endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string

Field	Description	Value Type
src_endpoint.owner.name	Recommended: The username. For example, janedoe1.	string
src_endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
src_endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
src_endpoint.owner.type_id	Recommended: The account type identifier.	integer
src_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string

Field	Description	Value Type
src_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint	Optional: The network proxy information pertaining to a specific endpoint.	object
src_endpoint.proxy_ endpoint.container	Recommended: The information describing an instance of a container.	object
src_endpoint.proxy_ endpoint.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container. For example: 13550340a8681c84c861a a....	object
src_endpoint.proxy_ endpoint.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
src_endpoint.proxy_ endpoint.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
src_endpoint.proxy_ endpoint.container.hash.value	Required: The digital fingerprint value.	string
src_endpoint.proxy_ endpoint.container.image	Recommended: The container image used as a template to run the container.	object
src_endpoint.proxy_ endpoint.container.image.name	Recommended: The image name. For example: elixir.	string

Field	Description	Value Type
src_endpoint.proxy_endpoint.container.image.uid	Required: The unique image ID. For example: 77af4d6b9913.	string
src_endpoint.proxy_endpoint.container.name	Recommended: The container name.	string
src_endpoint.proxy_endpoint.container.size	Recommended: The size of the container image.	long
src_endpoint.proxy_endpoint.container.uid	Recommended: The full container unique identifier for this instantiation of the container. For example: ac2ea168264a08f9....	string
src_endpoint.proxy_endpoint.hostname	Recommended: The fully qualified name of the endpoint.	string
src_endpoint.proxy_endpoint.instance_uid	Recommended: The unique identifier of a VM instance.	string
src_endpoint.proxy_endpoint.interface_name	Recommended: The name of the network interface (For example, eth2).	string
src_endpoint.proxy_endpoint.interface_uid	Recommended: The unique identifier of the network interface.	string
src_endpoint.proxy_endpoint.ip	Recommended: The IP address of the endpoint, in either IPv4 or IPv6 format.	string
src_endpoint.proxy_endpoint.name	Recommended: The short name of the endpoint.	string

Field	Description	Value Type
src_endpoint.proxy_ endpoint.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
src_endpoint.proxy_endpoint.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
src_endpoint.proxy_ endpoint.owner.account	Optional: The user's account or the account associated with the user.	object
src_endpoint.proxy_ endpoint.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
src_endpoint.proxy_ endpoint.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
src_endpoint.proxy_ endpoint.owner.account.type_id	Recommended: The normalized account type identifier.	integer
src_endpoint.proxy_ endpoint.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
src_endpoint.proxy_ endpoint.owner.domain	Optional: The domain where the user is defined. For example: the LDAP or Active Directory domain.	string
src_endpoint.proxy_ endpoint.owner.email_addr	Optional: The user's primary email address.	string

Field	Description	Value Type
src_endpoint.proxy_ endpoint.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
src_endpoint.proxy_ endpoint.owner.name	Recommended: The username. For example, janedoe1.	string
src_endpoint.proxy_ endpoint.owner.org	Optional: Organization and org unit related to the user.	object
src_endpoint.proxy_ endpoint.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
src_endpoint.proxy_ endpoint.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
src_endpoint.proxy_ endpoint.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
src_endpoint.proxy_ endpoint.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
src_endpoint.proxy_ endpoint.owner.type	Optional: The type of the user. For example, System, AWS IAM User,	string
src_endpoint.proxy_ endpoint.owner.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
src_endpoint.proxy_endpoint.owner.uid	Recommended: The unique user identifier. For example, the Windows user SID, ActiveDirectory DN or AWS user ARN.	string
src_endpoint.proxy_endpoint.port	Recommended: The port used for communication within the network connection.	integer
src_endpoint.proxy_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.proxy_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.proxy_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.svc_name	Recommended: The service name in service-to-service connections.	string
src_endpoint.type_id	Recommended: The network endpoint type ID.	integer
src_endpoint.uid	Recommended: The unique identifier of the endpoint.	string
src_endpoint.zone	Optional: The network zone or LAN segment.	string
start_time	Optional: The start time of a time period, or the time of the least recent event included in the aggregate event.	long

Field	Description	Value Type
status	Recommended: The event status, normalized to the caption of the status_id value.	string
status_code	Recommended: The event status code, as reported by the event source.	string
status_detail	Recommended: The status details contains additional information about the event/finding outcome.	string
status_id	Recommended: The normalized identifier of the event status.	integer
stratum	Recommended: The stratum level of the NTP server's time source, normalized to the caption of the stratum_id value.	string
stratum_id	Recommended: The normalized identifier of the stratum level, as defined in RFC-5905.	integer
timezone_offset	Recommended: The number of minutes that the reported event time is ahead or behind UTC.	integer
traffic	Optional: The network traffic refers to the amount of data moving across a network at a given point of time.	object

Field	Description	Value Type
traffic.bytes	Recommended: The total number of bytes (in and out).	long
traffic.bytes_in	Optional: The number of bytes sent from the destination to the source.	long
traffic.bytes_out	Optional: The number of bytes sent from the source to the destination.	long
traffic.chunks	Optional: The total number of chunks (in and out).	long
traffic.chunks_out	Optional: The number of chunks sent from the source to the destination.	long
traffic.packets	Recommended: The total number of packets (in and out).	long
traffic.packets_in	Optional: The number of packets sent from the destination to the source.	long
traffic.packets_out	Optional: The number of packets sent from the source to the destination.	long
transaction_uid	Recommended: The unique identifier of the transaction.	string
tree_uid	Recommended: The tree id is a unique SMB identifier which represents an open connection to a share.	string
tunnel_interface	Recommended: The information about the virtual tunnel interface.	object

Field	Description	Value Type
tunnel_interface.hostname	Recommended: The hostname associated with the network interface.	string
tunnel_interface.ip	Recommended: The IP address associated with the network interface.	string
tunnel_interface.mac	Recommended: The MAC address of the network interface.	string
tunnel_interface.name	Recommended: The name of the network interface.	string
tunnel_interface.type_id	Required: The network interface type identifier.	integer
tunnel_type	Recommended: The tunnel type. Example: Split or Full.	string
tunnel_type_id	Recommended: The normalized tunnel type ID.	integer
type	Recommended: The type of FTP network connection (For example, active, passive).	string
type_name	Optional: The event type name, as defined by the type_uid.	string
type_uid	Required: The event/finding type ID. It identifies the event's semantics and structure.	long
unmapped	Optional: The attributes that are not mapped to the event schema.	object

Field	Description	Value Type
url	Required: The URL included in the email content.	object
url.category_ids	Recommended: The Website categorization identifiers.	array of integers
url.hostname	Recommended: The URL host as extracted from the URL.	string
url.path	Recommended: The URL path as extracted from the URL.	string
url.port	Recommended: The URL port. For example: 80.	integer
url.query_string	Recommended: The query portion of the URL.	string
url.scheme	Recommended: The scheme portion of the URL.	string
url.url_string	Recommended: The URL string.	string
user	Recommended: The user associated with the tunnel activity.	object
user.account	Optional: The user's account or the account associated with the user.	object
user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string

Field	Description	Value Type
user.account.type_id	Recommended: The normalized account type identifier.	integer
user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
user.domain	Optional: The domain where the user is defined.	string
user.email_addr	Optional: The user's primary email address.	string
user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
user.name	Recommended: The username. For example, janedoe1.	string
user.org	Optional: Organization and org unit related to the user.	object
user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string

Field	Description	Value Type
user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
user.type	Optional: The type of the user. For example, System, AWS IAM User,	string
user.type_id	Recommended: The account type identifier.	integer
user.uid	Recommended: The unique user identifier.	string
version	Required: The version number of the NTP protocol.	string
volume	Required: The allocated event volume for the event in bytes.	long

OCSF System Activity Schema

The System Activity class captures events related to operating system and kernel-level operations, including process lifecycle events (creation, termination), file system operations, registry modifications, kernel module loading, and scheduled task execution. This schema is part of the OCSF (Open Cybersecurity Schema Framework) standard. For general OCSF guidance and field naming conventions, see "OCSF Schema Overview" on page 39.

For more information on the OCSF Version 1.2 standard, see OCSF v1.2 Documentation.

> **Note:** For OCSF schema fields, array indexes start at `array[1]` and not `array[0]`.

Field	Description	Value Type
access_mask	Recommended: The access mask in a platform-native format.	integer
action	Optional: The normalized caption of 'action_id' or the source-specific action.	string
action_id	Required: The normalized action taken by a control or other policy-based system leading to an outcome or disposition.	integer
activity_id	Required: The normalized identifier of the activity that triggered the event.	integer
activity_name	Optional: The event activity name, as defined by the activity_ id.	string
actor	Required: The actor process that loaded or unloaded the driver/extension.	object
actor.app_uid	Optional: The unique identifier of the client application or service that initiated the activity.	string
actor.idp	Optional: This object describes details about the Identity Provider used.	object

Field	Description	Value Type
actor.idp.name	Recommended: The name of the identity provider.	string
actor.idp.uid	Recommended: The unique identifier of the identity provider.	string
actor.invoked_by	Optional: The name of the service that invoked the activity as described in the event.	string
actor.process	Recommended: The process that initiated the activity.	object
actor.process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.container	Recommended: The information describing an instance of a container.	object
actor.process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container.	object
actor.process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string

Field	Description	Value Type
actor.process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.container.name	Recommended: The container name.	string
actor.process.container.size	Recommended: The size of the container image.	long
actor.process.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string
actor.process.created_time	Recommended: The time when the process was created/started.	long

Field	Description	Value Type
actor.process.file	Recommended: The process file object.	object
actor.process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
actor.process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.file.name	Required: The name of the file.	string

Field	Description	Value Type
actor.process.file.parent_folder	Optional: The parent folder in which the file resides.	string
actor.process.file.path	Recommended: The full path to the file.	string
actor.process.file.size	Optional: The size of data, in bytes.	long
actor.process.file.type	Optional: The file type.	string
actor.process.file.type_id	Required: The file type ID.	integer
actor.process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string

Field	Description	Value Type
actor.process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.container	Recommended: The information describing an instance of a container.	object
actor.process.parent_ process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container.	object
actor.process.parent_ process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_ process.container.hash.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
actor.process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string
actor.process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
actor.process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.file.name	Required: The name of the file.	string
actor.process.parent_process.file.path	Recommended: The full path to the file.	string
actor.process.parent_process.file.type	Optional: The file type.	string
actor.process.parent_process.file.type_id	Required: The file type ID.	integer

Field	Description	Value Type
actor.process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
actor.process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
actor.process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process	Recommended: The parent process of this process object.	object
actor.process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
actor.process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object

Field	Description	Value Type
actor.process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container.	object
actor.process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
actor.process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string
actor.process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
actor.process.parent_process.parent_process.container.name	Recommended: The container name.	string
actor.process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
actor.process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string

Field	Description	Value Type
actor.process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
actor.process.parent_process.parent_process.file	Recommended: The process file object.	object
actor.process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
actor.process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
actor.process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
actor.process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
actor.process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
actor.process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
actor.process.parent_process.parent_process.file.name	Required: The name of the file.	string

Field	Description	Value Type
actor.process.parent_process.parent_process.file.path	Recommended: The full path to the file.	string
actor.process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
actor.process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
actor.process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
actor.process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer

Field	Description	Value Type
actor.process.parent_process.parent_process.user.uid	Recommended: The unique user identifier.	string
actor.process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.parent_process.session	Optional: The user session under which this process is running.	object
actor.process.parent_process.session.created_time	Recommended: The time when the session was created.	long
actor.process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.parent_process.user	Recommended: The user under which this process is running.	object
actor.process.parent_process.user.domain	Optional: The domain where the user is defined.	string

Field	Description	Value Type
actor.process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.parent_process.user.type_id	Recommended: The account type identifier.	integer
actor.process.parent_process.user.uid	Recommended: The unique user identifier.	string
actor.process.pid	Recommended: The process identifier, as reported by the operating system.	integer
actor.process.session	Optional: The user session under which this process is running.	object
actor.process.session.created_time	Recommended: The time when the session was created.	long
actor.process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.process.session.issuer	Recommended: The identifier of the session issuer.	string
actor.process.session.uid	Recommended: The unique identifier of the session.	string
actor.process.session.uuid	Optional: The universally unique identifier of the session.	string

Field	Description	Value Type
actor.process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
actor.process.user	Recommended: The user under which this process is running.	object
actor.process.user.account	Optional: The user's account or the account associated with the user.	object
actor.process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.process.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.process.user.domain	Optional: The domain where the user is defined.	string
actor.process.user.email_addr	Optional: The user's primary email address.	string

Field	Description	Value Type
actor.process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.process.user.name	Recommended: The username. For example, `janedoe1`.	string
actor.process.user.org	Optional: Organization and org unit related to the user.	object
actor.process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string

Field	Description	Value Type
actor.process.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
actor.process.user.type_id	Recommended: The account type identifier.	integer
actor.process.user.uid	Recommended: The unique user identifier.	string
actor.session	Optional: The user session from which the activity was initiated.	object
actor.session.created_time	Recommended: The time when the session was created.	long
actor.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
actor.session.issuer	Recommended: The identifier of the session issuer.	string
actor.session.uid	Recommended: The unique identifier of the session.	string
actor.session.uuid	Optional: The universally unique identifier of the session.	string
actor.user	Recommended: The user that initiated the activity or the user context from which the activity was initiated.	object

Field	Description	Value Type
actor.user.account	Optional: The user's account or the account associated with the user.	object
actor.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
actor.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
actor.user.account.type_id	Recommended: The normalized account type identifier.	integer
actor.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
actor.user.domain	Optional: The domain where the user is defined.	string
actor.user.email_addr	Optional: The user's primary email address.	string
actor.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
actor.user.name	Recommended: The username. For example, `janedoe1`.	string

Field	Description	Value Type
actor.user.org	Optional: Organization and org unit related to the user.	object
actor.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
actor.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
actor.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
actor.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
actor.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
actor.user.type_id	Recommended: The account type identifier.	integer
actor.user.uid	Recommended: The unique user identifier.	string

Field	Description	Value Type
actual_permissions	Recommended: The permissions that were granted to the in a platform-native format.	integer
api	Optional: Describes details about a typical API (Application Programming Interface) call.	object
api.operation	Required: Verb/Operation associated with the request	string
api.request	Recommended: Details pertaining to the API request.	object
api.request.uid	Required: The unique request identifier.	string
api.response	Recommended: Details pertaining to the API response.	object
api.response.code	Recommended: The numeric response sent to a request.	integer
api.response.error	Recommended: Error Code	string
api.response.error_message	Recommended: Error Message	string

Field	Description	Value Type
api.response.message	Recommended: The description of the event/finding, as defined by the source.	string
api.service	Optional: The information pertaining to the API service.	object
api.service.name	Recommended: The name of the service.	string
api.service.uid	Recommended: The unique identifier of the service.	string
api.service.version	Recommended: The version of the service.	string
base_address	Recommended: The memory address that was access or requested.	string
category_name	Optional: The event category name, as defined by category_uid value.	string
category_uid	Required: The category unique identifier of the event.	integer
class_name	Optional: The event class name, as defined by the class_uid.	string

Field	Description	Value Type
class_uid	Required: The unique identifier of a class. A class describes the attributes available in an event.	integer
cloud	Required: Describes details about the Cloud environment where the event was originally created or logged.	object
cloud.account	Optional: The account object describes details about the account that was the source or target of the activity.	object
cloud.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
cloud.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
cloud.account.type_id	Recommended: The normalized account type identifier.	integer
cloud.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
cloud.org	Optional: Organization and org unit relevant to the event or object.	object

Field	Description	Value Type
cloud.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
cloud.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
cloud.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
cloud.provider	Required: The unique name of the Cloud services provider, such as AWS, MS Azure, GCP, etc.	string
cloud.region	Recommended: The name of the cloud region, as defined by the cloud provider.	string
component	Recommended: <p>The name or relative pathname of a sub-component of the data object, if applicable.	string

Field	Description	Value Type
count	Optional: The number of times that events in the same logical group occurred during the event Start Time to End Time period.	integer
create_mask	Recommended: The original Windows mask that is required to create the object.	string
device	Recommended: An addressable device, computer system or host.	object
device.container	Recommended: The information describing an instance of a container.	object
device.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container.	object
device.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
device.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
device.container.hash.value	Required: The digital fingerprint value.	string
device.container.image	Recommended: The container image used as a template to run the container.	object
device.container.image.name	Recommended: The image name. For example: `elixir`.	string
device.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
device.container.name	Recommended: The container name.	string
device.container.size	Recommended: The size of the container image.	long
device.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string
device.domain	Optional: The network domain where the device resides.	string
device.groups	Optional: The group names to which the device belongs. For example: `["Windows Laptops", "Engineering"]`.	array of objects

Field	Description	Value Type
device.groups.desc	Optional: The group description.	string
device.groups.domain	Optional: The domain where the group is defined.	string
device.groups.name	Recommended: The group name.	string
device.groups.type	Optional: The type of the group or account.	string
device.groups.uid	Recommended: The unique identifier of the group.	string
device.hostname	Recommended: The device hostname.	string
device.instance_uid	Recommended: The unique identifier of a VM instance.	string
device.interface_name	Recommended: The name of the network interface (For example, eth2).	string
device.interface_uid	Recommended: The unique identifier of the network interface.	string
device.ip	Recommended: The device IP address, in either IPv4 or IPv6 format.	string
device.is_compliant	Optional: The event occurred on a compliant device.	boolean

Field	Description	Value Type
device.is_managed	Optional: The event occurred on a managed device.	boolean
device.name	Recommended: The alternate device name, ordinarily as assigned by an administrator.	string
device.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
device.os	Optional: The endpoint operating system.	object
device.os.name	Required: The operating system name.	string
device.os.type	Optional: The type of the operating system.	string
device.os.type_id	Required: The type identifier of the operating system.	integer
device.owner	Recommended: The identity of the service or user account that owns the endpoint or was last logged into it.	object
device.owner.account	Optional: The user's account or the account associated with the user.	object

Field	Description	Value Type
device.owner.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
device.owner.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string
device.owner.account.type_id	Recommended: The normalized account type identifier.	integer
device.owner.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
device.owner.domain	Optional: The domain where the user is defined.	string
device.owner.email_addr	Optional: The user's primary email address.	string
device.owner.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
device.owner.name	Recommended: The username. For example, `janedoe1`.	string
device.owner.org	Optional: Organization and org unit related to the user.	object

Field	Description	Value Type
device.owner.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
device.owner.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string
device.owner.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
device.owner.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
device.owner.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
device.owner.type_id	Recommended: The account type identifier.	integer
device.owner.uid	Recommended: The unique user identifier.	string
device.region	Recommended: The region where the virtual machine is located. For example, an AWS Region.	string

Field	Description	Value Type
device.type	Optional: The device type.	string
device.type_id	Required: The device type ID.	integer
device.uid	Recommended: The unique identifier of the device.	string
driver	Required: The driver that was loaded/unloaded into the kernel	object
driver.file	Required: The driver/extension file object.	object
driver.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
driver.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
driver.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
driver.file.hashes	Recommended: An array of hash attributes.	array of objects

Field	Description	Value Type
driver.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
driver.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
driver.file.hashes.value	Required: The digital fingerprint value.	string
driver.file.name	Required: The name of the file.	string
driver.file.parent_folder	Optional: The parent folder in which the file resides.	string
driver.file.path	Recommended: The full path to the file.	string
driver.file.size	Optional: The size of data, in bytes.	long
driver.file.type	Optional: The file type.	string
driver.file.type_id	Required: The file type ID.	integer
duration	Optional: The event duration or aggregate time, the amount of time the event covers from `start_time` to `end_time` in milliseconds.	integer

Field	Description	Value Type
end_time	Optional: The end time of a time period, or the time of the most recent event included in the aggregate event.	long
event_seq_id	Required: The sequence identifier for the event.	long
event_size	Required: The size of the event in bytes.	long
event_time	Required: The normalized event occurrence time or the finding creation time.	long
exit_code	Recommended: The exit code reported by a process when it terminates.	integer
file	Required: The file that is the target of the activity.	object
file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer

Field	Description	Value Type
file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
file.hashes	Recommended: An array of hash attributes.	array of objects
file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
file.hashes.value	Required: The digital fingerprint value.	string
file.name	Required: The name of the file.	string
file.parent_folder	Optional: The parent folder in which the file resides.	string
file.path	Recommended: The full path to the file.	string
file.size	Optional: The size of data, in bytes.	long
file.type	Optional: The file type.	string
file.type_id	Required: The file type ID.	integer

Field	Description	Value Type
file_diff	Recommended: File content differences used for change detection.	string
file_result	Recommended: The resulting file object when the activity was allowed and successful.	object
file_result.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
file_result.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
file_result.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
file_result.hashes	Recommended: An array of hash attributes.	array of objects
file_result.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string

Field	Description	Value Type
file_result.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
file_result.hashes.value	Required: The digital fingerprint value.	string
file_result.name	Required: The name of the file.	string
file_result.parent_folder	Optional: The parent folder in which the file resides.	string
file_result.path	Recommended: The full path to the file.	string
file_result.size	Optional: The size of data, in bytes.	long
file_result.type	Optional: The file type.	string
file_result.type_id	Required: The file type ID.	integer
indexed_time	Required: The time when the event was indexed.	long
injection_type	Recommended: The process injection method, normalized to the caption of the injection_type_id value.	string
injection_type_id	Recommended: The normalized identifier of the process injection method.	integer

Field	Description	Value Type
job	Required: The job object that pertains to the event.	object
job.cmd_line	Recommended: The job command line.	string
job.created_time	Recommended: The time when the job was created.	long
job.desc	Recommended: The description of the job.	string
job.file	Required: The file that pertains to the job.	object
job.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
job.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
job.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
job.file.hashes	Recommended: An array of hash attributes.	array of objects

Field	Description	Value Type
job.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
job.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
job.file.hashes.value	Required: The digital fingerprint value.	string
job.file.name	Required: The name of the file.	string
job.file.parent_folder	Optional: The parent folder in which the file resides.	string
job.file.path	Recommended: The full path to the file.	string
job.file.size	Optional: The size of data, in bytes.	long
job.file.type	Optional: The file type.	string
job.file.type_id	Required: The file type ID.	integer
job.last_run_time	Recommended: The time when the job was last run.	long
job.name	Required: The name of the job.	string

Field	Description	Value Type
job.run_state_id	Recommended: The run state ID of the job.	integer
kernel	Required: The target kernel resource.	object
kernel.name	Required: The name of the kernel resource.	string
kernel.type_id	Required: The type of the kernel resource.	integer
last_run_time	Recommended: The prefetch file last run time.	long
message	Recommended: The description of the event/finding, as defined by the source.	string
metadata	Required: The metadata associated with the event or a finding.	object
metadata.correlation_uid	Optional: The unique identifier used to correlate events.	string
metadata.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
metadata.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.event_code	Optional: The Event ID or Code that the product uses to describe the event.	string
metadata.log_level	Optional: The audit level at which an event was generated.	string
metadata.log_name	Recommended: The event log name.	string
metadata.log_provider	Recommended: The logging provider or logging service that logged the event.	string
metadata.original_time	Recommended: The original event time as reported by the event source.	string
metadata.product	Required: The product that reported the event.	object

Field	Description	Value Type
metadata.product.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
metadata.product.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
metadata.product.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
metadata.product.feature	Optional: The feature that reported the event.	object
metadata.product.feature.name	Recommended: The name of the feature.	string
metadata.product.feature.uid	Recommended: The unique identifier of the feature.	string
metadata.product.feature.version	Recommended: The version of the feature.	string
metadata.product.name	Recommended: The name of the product.	string
metadata.product.uid	Recommended: The unique identifier of the product.	string
metadata.product.vendor_name	Required: The name of the vendor of the product.	string

Field	Description	Value Type
metadata.product.version	Recommended: The version of the product, as defined by the event source.	string
metadata.profiles	Optional: The list of profiles used to create the event.	array of strings
metadata.sequence	Optional: Sequence number of the event.	integer
metadata.tenant_uid	Recommended: The unique tenant identifier.	string
metadata.uid	Optional: The logging system-assigned unique identifier of an event instance.	string
metadata.version	Required: The version of the OCSF schema, using Semantic Versioning Specification.	string
module	Required: The module that was loaded or unloaded.	object
module.base_address	Recommended: The memory address where the module was loaded.	string
module.file	Recommended: The module file object.	object

Field	Description	Value Type
module.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
module.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
module.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
module.file.hashes	Recommended: An array of hash attributes.	array of objects
module.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
module.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
module.file.hashes.value	Required: The digital fingerprint value.	string
module.file.name	Required: The name of the file.	string

Field	Description	Value Type
module.file.parent_folder	Optional: The parent folder in which the file resides.	string
module.file.path	Recommended: The full path to the file.	string
module.file.size	Optional: The size of data, in bytes.	long
module.file.type	Optional: The file type.	string
module.file.type_id	Required: The file type ID.	integer
module.load_type_id	Required: The normalized identifier of the load type. It identifies how the module was loaded in memory.	integer
module.start_address	Recommended: The start address of the execution.	string
module.type	Recommended: The module type.	string
name	Required: The name of the prefetch file that is the target of the query.	string
observables	Recommended: The observables associated with the event or a finding.	array of objects

Field	Description	Value Type
observables.name	Required: The full name of the observable attribute. The `name` is a pointer/reference to an attribute within the event data. For example: `file.name`.	string
observables.type	Optional: The observable value type name.	string
observables.type_id	Required: The observable value type identifier.	integer
observables.value	Optional: The value associated with the observable attribute. The meaning of the value depends on the observable type.	string
open_mask	Recommended: The Windows options needed to open a registry key.	integer
original_event_time	Required: The original event time from the source system.	long
prev_reg_key	Recommended: The registry key before the mutation	object
prev_reg_key.path	Required: The full path to the registry key.	string

Field	Description	Value Type
process	Required: The process that had memory allocated, read/written, or had other manipulation activities performed on it.	object
process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
process.container	Recommended: The information describing an instance of a container.	object
process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container.	object
process.container.hash.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.container.hash.value	Required: The digital fingerprint value.	string

Field	Description	Value Type
process.container.image	Recommended: The container image used as a template to run the container.	object
process.container.image.name	Recommended: The image name. For example: `elixir`.	string
process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
process.container.name	Recommended: The container name.	string
process.container.size	Recommended: The size of the container image.	long
process.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string
process.created_time	Recommended: The time when the process was created/started.	long
process.file	Recommended: The process file object.	object
process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object

Field	Description	Value Type
process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
process.file.hashes	Recommended: An array of hash attributes.	array of objects
process.file.hashes.algorithm	Optional: The hash algorithm used to create the digital fingerprint, normalized to the caption of 'algorithm_id'.	string
process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.file.hashes.value	Required: The digital fingerprint value.	string
process.file.name	Required: The name of the file.	string
process.file.parent_folder	Optional: The parent folder in which the file resides.	string
process.file.path	Recommended: The full path to the file.	string
process.file.size	Optional: The size of data, in bytes.	long

Field	Description	Value Type
process.file.type	Optional: The file type.	string
process.file.type_id	Required: The file type ID.	integer
process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
process.parent_process	Recommended: The parent process of this process object.	object
process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string

Field	Description	Value Type
process.parent_process.container	Recommended: The information describing an instance of a container.	object
process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container.	object
process.parent_ process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.parent_ process.container.hash.value	Required: The digital fingerprint value.	string
process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
process.parent_ process.container.image.name	Recommended: The image name. For example: `elixir`.	string
process.parent_ process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
process.parent_process.container.name	Recommended: The container name.	string
process.parent_process.container.size	Recommended: The size of the container image.	long

Field	Description	Value Type
process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string
process.parent_process.created_time	Recommended: The time when the process was created/started.	long
process.parent_process.file	Recommended: The process file object.	object
process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer

Field	Description	Value Type
process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
process.parent_process.file.name	Required: The name of the file.	string
process.parent_process.file.path	Recommended: The full path to the file.	string
process.parent_process.file.type	Optional: The file type.	string
process.parent_process.file.type_id	Required: The file type ID.	integer
process.parent_process.integrity	Optional: The process integrity level, normalized to the caption of the direction_id value. In the case of 'Other', it is defined by the event source (Windows only).	string
process.parent_process.integrity_id	Optional: The normalized identifier of the process integrity level (Windows only).	integer
process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer

Field	Description	Value Type
process.parent_process.parent_process	Recommended: The parent process of this process object.	object
process.parent_process.parent_process.cmd_line	Recommended: The full command line used to launch an application, service, process, or job.	string
process.parent_process.parent_process.container	Recommended: The information describing an instance of a container.	object
process.parent_process.parent_process.container.hash	Recommended: Commit hash of image created for docker or the SHA256 hash of the container.	object
process.parent_process.parent_process.container.hash.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.parent_process.parent_process.container.hash.value	Required: The digital fingerprint value.	string
process.parent_process.parent_process.container.image	Recommended: The container image used as a template to run the container.	object
process.parent_process.parent_process.container.image.name	Recommended: The image name. For example: `elixir`.	string

Field	Description	Value Type
process.parent_process.parent_process.container.image.uid	Required: The unique image ID. For example: `77af4d6b9913`.	string
process.parent_process.parent_process.container.name	Recommended: The container name.	string
process.parent_process.parent_process.container.size	Recommended: The size of the container image.	long
process.parent_process.parent_process.container.uid	Recommended: The full container unique identifier for this instantiation of the container.	string
process.parent_process.parent_process.created_time	Recommended: The time when the process was created/started.	long
process.parent_process.parent_process.file	Recommended: The process file object.	object
process.parent_process.parent_process.file.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
process.parent_process.parent_process.file.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
process.parent_process.parent_process.file.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer

Field	Description	Value Type
process.parent_process.parent_process.file.hashes	Recommended: An array of hash attributes.	array of objects
process.parent_process.parent_process.file.hashes.algorithm_id	Required: The identifier of the normalized hash algorithm, which was used to create the digital fingerprint.	integer
process.parent_process.parent_process.file.hashes.value	Required: The digital fingerprint value.	string
process.parent_process.parent_process.file.name	Required: The name of the file.	string
process.parent_process.parent_process.file.path	Recommended: The full path to the file.	string
process.parent_process.parent_process.file.type_id	Required: The file type ID.	integer
process.parent_process.parent_process.name	Recommended: The friendly name of the process, for example: `Notepad++`.	string
process.parent_process.parent_process.namespace_pid	Recommended: If running under a process namespace (such as in a container), the process identifier within that process namespace.	integer
process.parent_process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer

Field	Description	Value Type
process.parent_process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
process.parent_process.parent_process.user	Recommended: The user under which this process is running.	object
process.parent_process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
process.parent_process.parent_process.user.type_id	Recommended: The account type identifier.	integer
process.parent_process.parent_process.user.uid	Recommended: The unique user identifier.	string
process.parent_process.pid	Recommended: The process identifier, as reported by the operating system.	integer
process.parent_process.session	Optional: The user session under which this process is running.	object
process.parent_process.session.created_time	Recommended: The time when the session was created.	long
process.parent_process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
process.parent_process.session.issuer	Recommended: The identifier of the session issuer.	string

Field	Description	Value Type
process.parent_process.session.uid	Recommended: The unique identifier of the session.	string
process.parent_process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
process.parent_process.user	Recommended: The user under which this process is running.	object
process.parent_process.user.domain	Optional: The domain where the user is defined.	string
process.parent_process.user.name	Recommended: The username. For example, `janedoe1`.	string
process.parent_process.user.type_id	Recommended: The account type identifier.	integer
process.parent_process.user.uid	Recommended: The unique user identifier.	string
process.pid	Recommended: The process identifier, as reported by the operating system.	integer
process.session	Optional: The user session under which this process is running.	object
process.session.created_time	Recommended: The time when the session was created.	long

Field	Description	Value Type
process.session.is_remote	Recommended: The indication of whether the session is remote.	boolean
process.session.issuer	Recommended: The identifier of the session issuer.	string
process.session.uid	Recommended: The unique identifier of the session.	string
process.session.uuid	Optional: The universally unique identifier of the session.	string
process.uid	Recommended: A unique identifier for this process assigned by the producer (tool).	string
process.user	Recommended: The user under which this process is running.	object
process.user.account	Optional: The user's account or the account associated with the user.	object
process.user.account.name	Recommended: The name of the account (For example, GCP Account Name).	string
process.user.account.type	Optional: The account type, normalized to the caption of 'account_type_id'.	string

Field	Description	Value Type
process.user.account.type_id	Recommended: The normalized account type identifier.	integer
process.user.account.uid	Recommended: The unique identifier of the account (For example, AWS Account ID).	string
process.user.domain	Optional: The domain where the user is defined.	string
process.user.email_addr	Optional: The user's primary email address.	string
process.user.full_name	Optional: The full name of the person, as per the LDAP Common Name attribute (cn).	string
process.user.name	Recommended: The username. For example, `janedoe1`.	string
process.user.org	Optional: Organization and org unit related to the user.	object
process.user.org.name	Recommended: The name of the organization. For example, Widget, Inc.	string
process.user.org.ou_name	Recommended: The name of the organizational unit, within an organization. For example, Finance, IT, R&D	string

Field	Description	Value Type
process.user.org.uid	Recommended: The unique identifier of the organization. For example, its Active Directory or AWS Org ID.	string
process.user.risk_level	Optional: The risk level, normalized to the caption of the risk_level_id value.	string
process.user.type	Optional: The type of the user. For example, System, AWS IAM User, etc.	string
process.user.type_id	Recommended: The account type identifier.	integer
process.user.uid	Recommended: The unique user identifier.	string
raw_data	Optional: The raw event/finding data as received from the source.	string
raw_data_size	Optional: The size of the raw data which was transformed into an OCSF event, in bytes.	long
reg_key	Required: The registry key.	object
reg_key.path	Required: The full path to the registry key.	string

Field	Description	Value Type
reg_value	Required: The registry value.	object
reg_value.name	Required: The name of the registry value.	string
reg_value.path	Required: The full path to the registry key, where the value is located.	string
reg_value.type_id	Recommended: The value type ID.	integer
requested_permissions	Recommended: The permissions mask that were requested by the process.	integer
run_count	Recommended: The prefetch file run count.	integer
severity	Optional: The event severity, normalized to the caption of the severity_id value.	string
severity_id	Required: <p>The normalized identifier of the event/finding severity.	integer
start_time	Optional: The start time of a time period, or the time of the least recent event included in the aggregate event.	long

Field	Description	Value Type
status	Recommended: The event status, normalized to the caption of the status_id value.	string
status_code	Recommended: The event status code, as reported by the event source.	string
status_detail	Recommended: The status details contains additional information about the event/finding outcome.	string
status_id	Recommended: The normalized identifier of the event status.	integer
timezone_offset	Recommended: The number of minutes that the reported event `time` is ahead or behind UTC.	integer
type_name	Optional: The event type name, as defined by the type_uid.	string
type_uid	Required: The event/finding type ID.	long
unmapped	Optional: The attributes that are not mapped to the event schema.	object
volume	Required: The allocated event volume for the event in bytes.	long

Field	Description	Value Type
win_resource	Required: The Windows resource object that was accessed, such as a mutant or timer.	object
win_resource.data_classification	Recommended: The Data Classification object includes information about data classification levels and data category types.	object
win_resource.data_classification.category_id	Recommended: The normalized identifier of the data classification category.	integer
win_resource.data_classification.confidentiality_id	Recommended: The normalized identifier of the file content confidentiality indicator.	integer
win_resource.type_id	Required: The normalized type identifier of the Windows resource object accessed.	integer

OCSF Correlation Tables

The Anomali Security Analytics correlation engine generates OCSF-formatted output tables when it processes your event logs against ThreatStream intelligence.

The three following tables span the OCSF schema for correlations and are available for querying with AQL in Anomali Search:

- iocmatch_ocsf: Stores the results of IOC correlations.
- dgamatch_ocsf: Stores the results of domain generated algorithm (DGA)

detections.

- tmmatch_ocsf: Store the results of threat model matching.

See "OCSF Schema Overview" on page 39 for more information on the OCSF Version 1.2 field names and conventions.

dgamatch_ocsf

The dgamatch_ocsf lookup table is generated by the Anomali Security Analytics domain generated algorithm (DGA) detection feature and stores DGA detection results in the OCSF schema format. The DGA detection feature correlates domain names appearing in event logs with Anomali DGA intelligence.

See "OCSF Schema Overview" on page 39 for more information.

Field	Value Type	Description	Example
_date	string	Partition key, format YYYY-MM-DD.	2026-04-28
action	string	Caption of action_id.	Allowed, Blocked
action_id	number	Normalized action taken by a control or policy.	1
activity_id	number	Normalized activity that triggered the event.	1
activity_name	string	Caption of activity_id.	Create
actor	struct	Actor that performed the activity.	{"user": {"name": "john.smith@example.com", "uid": "101234567_15600_a1b2c3d4-5678-90ab-cdef-1234567890ab"}}

Field	Value Type	Description	Example
answers	list	DNS answers.	[{"hostname": "93.184.216.34", "type": "A"}]
api	struct	API-call details.	{"operation": "GetObject", "service": {"name": "s3"}}
category_name	string	Caption of category_uid.	Application Activity
category_uid	number	OCSF category identifier.	6
class_name	string	Caption of class_uid.	API Activity
class_uid	number	OCSF class identifier that describes the attribute shape of the event.	6003
cloud	struct	Cloud environment where the event originated.	{"provider": "AWS", "region": "us-east-1"}
connection_info	struct	Network connection protocol, direction, and related details.	{"protocol": "TCP", "direction": "outbound"}
count	number	Number of logically-grouped events in the aggregate.	23
device	struct	Addressable device or host.	{"hostname": "workstation01", "ip": "192.168.1.10"}

Field	Value Type	Description	Example
dga_prob	double	DGA probability score of the domain.	0.997
dga_tag	string	Additional comments and context for the DGA detection.	malicious
disposition	string	Caption of disposition_id.	Blocked
disposition_id	number	Outcome taken by a security control.	2
dst_endpoint	struct	Destination network endpoint.	{"svc_name": "OneDrive Site Collection a1b2c3d4-e5f6-7890-abcd-ef1234567890"}
duration	number	end_time minus start_time in milliseconds.	1000
email	struct	Email object.	{"from": "sender@example.com", "subject": "Invoice"}
end_time	number	End of an aggregated event window, in UNIX timestamp milliseconds.	1705557026030
enrichments	list	External-source enrichments attached to the event.	[{"name": "GeoIP", "value": "US"}]
event_seq_id	number	Event sequence identifier.	79618005142

Field	Value Type	Description	Example
event_size	number	Event size in bytes.	6338
event_time	number	Normalized event occurrence time, in UNIX timestamp milliseconds.	1777386930000
file	struct	Target file object.	{"name": "malware.exe", "path": "C:\\Windows\\Temp"}
http_request	struct	HTTP request details.	{"http_headers": null, "http_method": null, "user_agent": "Microsoft SkyDriveSync 25.206.1021.0003 ship; Windows NT 10.0 (26100)"}
indexed_time	number	When the event was indexed, in UNIX timestamp milliseconds.	1777498532359
indicator	string	Value of the DGA indicator (domain).	aksjfhalskdfhalskdjfh.com
ioc_status	string	Status assigned to the indicator.	active, inactive, falsepos
ioc_type	string	Indicator value type.	domain
malware_family	string	Group of applications with similar attack techniques.	Corebot, Gameover_DGA, Gameover_P2P, Hesperbot, MadMax, Necurs, QakBot, Recurs
message	string	Event description from the source.	DGA domain detected in DNS query

Field	Value Type	Description	Example
metadata	struct	Event metadata including product, log source, correlation uid, and data classification.	{"correlation_uid": null, "data_classification": {...}, "product": {"name": "Microsoft Defender for Cloud Apps", "vendor_name": "Microsoft"}, "version": "1.1.0"}
observables	list	Observables tied to the event.	[{"name": "randomdomain123.com", "type": "Hostname"}]
original_event_time	number	Original timestamp from the source system, in UNIX timestamp milliseconds.	1763233459000
query	struct	DNS query details.	{"hostname": "randomdomain123.com", "type": "A"}
raw_data	string	Raw event data from the source.	{"_id": "101234567_15600_a1b2c3d4-5678-90ab-cdef-1234567890ab", "appId": 15600, "appName": "Microsoft OneDrive for Business", "classifications": ["file"], "confidenceLevel": 20}
raw_data_size	number	Size of the original raw data in bytes.	4527
session	struct	Authenticated session details.	{"uid": "abc123"}
severity	string	Caption of severity_id.	Informational

Field	Value Type	Description	Example
severity_id	number	Normalized severity (lower = less severe).	1
src_endpoint	struct	Source network endpoint.	{"ip": "192.0.2.15"}
start_time	number	Start of an aggregated event window, in UNIX timestamp milliseconds.	1705557025030
status	string	Caption of status_ id.	Success, Failure
status_code	string	Source-reported status code.	200, 404
status_detail	string	Additional outcome details.	Connection established
status_id	number	Normalized status identifier.	1
timezone_ offset	number	Source timezone offset in minutes.	-480
type_name	string	Caption of type_ uid.	Detection Finding: Allowed
type_uid	number	Computed as class_uid × 100 + activity_id.	600301
unmapped	string	Source-specific attributes not mapped to OCSF.	{'RecordType': NULL, 'rawDataJson': {'Operation': ManagedSyncClientAllowed}, 'tranport': NULL}

Field	Value Type	Description	Example
user	struct	User being discovered or targeted.	{"name": "john.doe", "email": "john.doe@example.com"}
volume	number	Allocated event volume in bytes.	792007342

iocmatch_ocsf

The iocmatch_ocsf lookup table is generated by the Anomali Security Analytics IOC correlation engine and stores correlated event data in the OCSF schema format. The IOC correlation engine correlates your event logs with your ThreatStream IOCs.

See "OCSF Schema Overview" on page 39 for more information.

Field	Value Type	Description	Example
_date	string	Partition key, format YYYY-MM-DD.	2024-01-18
action	string	Caption of action_id.	Allowed, Blocked
action_id	number	Normalized action taken by a control or policy.	1
activity_id	number	Normalized activity that triggered the event.	6
activity_name	string	Caption of activity_id.	Allowed

Field	Value Type	Description	Example
actor	object	Actor that performed the activity.	{"user": {"name": "admin"}}
actor_ids	list	Associated actor IDs in ThreatStream.	[2342, 2313]
age	number	Event time minus intel first seen.	49
answers	list	DNS answers.	[{"hostname": "93.184.216.34", "type": "A"}]
api	object	API-call details.	{"operation": "GetObject", "service": {"name": "s3"}}
asn	string	Autonomous System Number.	15802
attackpattern_ids	list	Associated attack pattern IDs in ThreatStream.	[2342, 2313]
campaign_ids	list	Associated campaign IDs in ThreatStream.	[2342, 2313]
category_name	string	Caption of category_uid.	Network Activity
category_uid	number	OCSF category identifier.	4
class_name	string	Caption of class_uid.	Network Activity

Field	Value Type	Description	Example
class_uid	number	OCSF class identifier that describes the attribute shape of the event.	4001
classification	string	Classification of an intel.	private
cloud	object	Cloud environment where the event originated.	{"provider": "AWS", "region": "us-east-1"}
confidence	number	Confidence score assigned to the indicator.	75
connection_info	object	Network connection protocol, direction, and related details.	{"direction": "incoming", "direction_id": 1, "protocol_num": 6, "session": {"uid": "82791871"}}
count	number	Number of logically-grouped events in the aggregate.	23
country	string	Country associated with the indicator.	US
created_ts	number	First seen on ThreatStream, in UNIX timestamp milliseconds.	1774382999000
device	object	Addressable device or host.	{"hostname": "workstation01", "ip": "192.168.1.10"}

Field	Value Type	Description	Example
disposition	string	Caption of disposition_id.	Blocked
disposition_id	number	Outcome taken by a security control.	2
dst_endpoint	object	Destination network endpoint.	{"interface_name": "wan2", "ip": "1.2.3.4", "port": 52058}
duration	number	end_time minus start_time in milliseconds.	125
email	object	Email object.	{"from": "sender@example.com", "subject": "Invoice"}
end_time	number	End of an aggregated event window, in UNIX timestamp milliseconds.	1705557026030
enrichments	list	External-source enrichments attached to the event.	[{"name": "GeoIP", "value": "US"}]
event_seq_id	number	Event sequence identifier.	79618005142
event_size	number	Event size in bytes.	1821

Field	Value Type	Description	Example
event_time	number	Normalized event occurrence time, in UNIX timestamp milliseconds.	1778696793119
feed_id	number	Source feed ID in ThreatStream.	0
feed_name	string	Source of the indicator.	johndoe@acme.com
file	object	Target file object.	{"name": "malware.exe", "path": "C:\\Windows\\Temp"}
http_request	object	HTTP request details.	{"method": "GET", "url": "https://example.com"}
id	number	ID of an indicator.	41008949
import_ session_id	number	Import session ID.	93821
indexed_time	number	When the event was indexed, in UNIX timestamp milliseconds.	1778696793119
indicator	string	Value of an indicator.	50.207.148.140
intel_severity	string	Severity assigned to the indicator.	low, medium, high, very-high
ioc_match_id	string	Unique match ID, format {orgid}_{event_ id}_0_{ioc_id}.	1773_79618005142_0_ 59324653376

Field	Value Type	Description	Example
ioc_status	string	Status assigned to the indicator.	active, inactive, falsepos
ioc_type	string	Indicator value type.	ip, domain, url, email, md5
itype	string	Indicator type.	mal_ip
latitude	double	Latitude of the IP address.	29.397
longitude	double	Longitude of the IP address.	-94.92
message	string	Event description from the source.	Malicious IP matched IOC indicator
metadata	object	Event metadata including product, log source, correlation uid, and data classification.	{"correlation_uid": null, "data_classification": {...}, "product": {"name": "Fortigate Firewall", "vendor_name": "Fortinet"}, "version": "1.1.0"}
modified_ts	number	Last updated on ThreatStream, in UNIX timestamp milliseconds.	1774384127000
observables	list	Observables tied to the event.	[{"name": "evil.com", "type": "Hostname"}]
org	string	Registered owner (org) of the IP.	Comcast Business

Field	Value Type	Description	Example
original_event_time	number	Original timestamp from the source system, in UNIX timestamp milliseconds.	1778696340000
originator	number	Network directionality.	0 (inbound), 1 (outbound)
query	object	DNS query details.	{"hostname": "example.com", "type": "A"}
raw_data	string	Raw event data from the source.	date=2021-11-30 time=22:22:24 devname="nap9-alex00va-100d-1" type="utm" subtype="ips" level="alert" srcip=50.207.148.140 dstip=221.223.35.118 action="dropped" proto=6 attack="TCP.Bad.Flags"
raw_data_size	number	Size of the original raw data in bytes.	760
resource_uri	string	URI for the indicator.	/api/v1/intelligence/41008949
session	object	Authenticated session details.	{"uid": "abc123"}
severity	string	Caption of severity_id.	Low, Medium, High, Critical
severity_id	number	Normalized severity (lower = less severe).	3

Field	Value Type	Description	Example
src_endpoint	object	Source network endpoint.	{"ip": "192.168.1.10", "port": 443}
start_time	number	Start of an aggregated event window, in UNIX timestamp milliseconds.	1705557025030
status	string	Caption of status_id.	Success, Failure
status_code	string	Source-reported status code.	200, 404
status_detail	string	Additional outcome details.	Connection established
status_id	number	Normalized status identifier.	1
tags	list	Additional comments and context.	["popularity=low", "malware"]
timezone_offset	number	Source timezone offset in minutes.	-480
tipreport_ids	list	Associated threat bulletin IDs in ThreatStream.	[2342, 2313]
trusted_circle_ids	list	Trusted-circle IDs assigned to the indicator.	[1, 45]
trusted_circle_names	list	Trusted-circle names.	["circle1", "circle45"]

Field	Value Type	Description	Example
ttp_ids	list	Associated TTP IDs in ThreatStream.	[5473, 34724]
type_name	string	Caption of type_uid.	Detection Finding: Allowed
type_uid	number	Computed as class_uid × 100 + activity_id.	400101
unmapped	string	Source-specific attributes not mapped to OCSF.	{"custom_field": "custom_value"}
user	struct	User being discovered or targeted.	{"name": "john.doe", "email": "john.doe@example.com"}
volume	number	Allocated event volume in bytes.	2048
vulnerability_ids	list	Associated vulnerability IDs in ThreatStream.	[5473, 34724]

tmmatch_ocsf

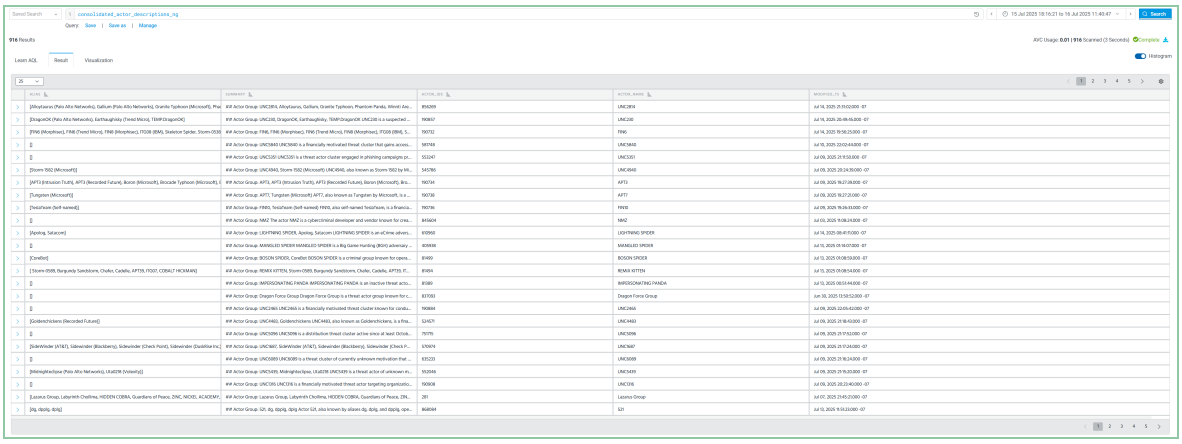
The tmmatch_ocsf lookup table is generated by the Anomali Security Analytics threat model correlation feature and stores threat model match data in the OCSF schema format. The threat model correlation feature produces a summary of threat model matches found in your event logs.

See "OCSF Schema Overview" on page 39 for more information.

Field	Value Type	Description	Example
_date	string	Partition key, format YYYY-MM-DD.	2024-01-18
event_time	number	Event time on the originating device, in UNIX timestamp milliseconds.	1705557025030
indexed_time	number	Time when the event was received by Match, in UNIX timestamp milliseconds.	1705557025030
ioc_match_id	string	Unique match ID linking to the corresponding iocmatch_ocsf row, format {orgid}_{event_id}_0_{ioc_id}.	1773_79618005142_0_59324653376
tm_id	number	Threat model ID in ThreatStream.	8472
tm_match_id	string	Unique ID for the threat model match, format {orgid}_{event_id}_0_{ioc_id}_{tm_id}.	1773_79618005142_0_59324653376_8472
tm_name	string	Threat model name in ThreatStream.	APT29 Campaign
tm_type	string	Threat model type in ThreatStream.	actor, campaign, ttp

consolidated_actor_descriptions_ng

The `consolidated_actor_descriptions_ng` search table is one of the data sources for the AI Actor Profile dashboard. It adds an AI description for every unique threat actor and its aliases on ThreatStream. The table is read-only and not subject to modification.



Field	Value Type	Description	Example
actor_ids	string	ID of the Actor	856269
actor_ name	string	Name of the Actor	Dragon Force Group
alias	array of strings	Other names associated with the Actor.	[Apolog, Satacom]

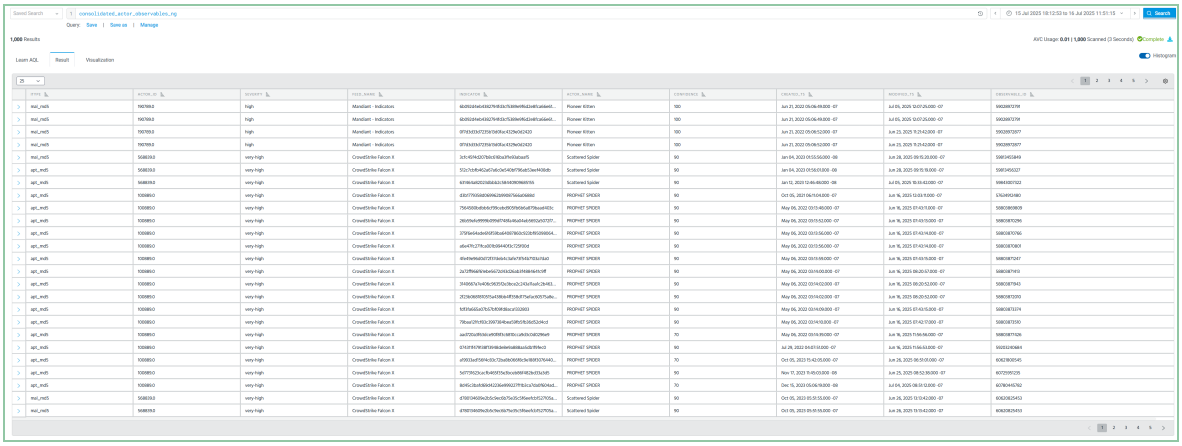
Field	Value Type	Description	Example
modified_ts	date	Time stamp of when the Actor was last updated in ThreatStream. Date is specified in the following format: `MMM dd, yyyy HH:mm:ss.SSS Z` where • `MMM` - month name • `dd` - day of the month • `yyyy` - year • `HH` - hour (24-clock, 2 digits) • `mm` - minutes • `ss` - seconds • `.SSS` - milliseconds (3 digits) • `Z` - timezone offset from UTC	Jun 30, 2025 13:50:52.000 -07

Field	Value Type	Description	Example
summary	string	Summary of the Actor Group.	## Actor Group: UNC2814, Alloytaurus, Gallium, Granite Typhoon, Phantom Panda, Winnti Area 6 UNC2814 is a suspected Chinese cyber espionage group confirmed by multiple cybersecurity firms including Palo Alto Networks, Microsoft, CrowdStrike, and ProtectWise. Active since at least 2011, the group primarily targets international governments and telecommunications organizations with motivations centered on espionage. UNC2814 has merged with UNC4512 as of September 2024. The group employs a wide range of malware such as CHINACHOP, COLDEYE (including a Linux variant), COLDNOSE, HALFSPOT, HTRAN, POISONIVY, TROCHILUS, and ZXSHELL, alongside tools like WHOAMI. Their operations originate from China, specifically the Asia region and East Asia subregion. UNC2814 utilizes diverse tactics and techniques mapped to MITRE ATT&CK, including resource development (compromise infrastructure, develop and obtain capabilities, install digital certificates), initial access (external remote services, exploit public-facing applications), execution (command and scripting interpreters), persistence (external remote services, server software components, web shells, boot or

Field	Value Type	Description	Example
			logon autostart execution), privilege escalation (process injection, hijack execution flow), defense evasion (rootkits, obfuscation, process injection, indicator removal, virtualization/sandbox evasion, code signing), credential access (OS credential dumping), discovery (system services, network configuration, process and file discovery), collection (archiving collected data), command and control (application layer protocols, web protocols, encrypted channels), ingress tool transfer, data encoding, and impact (system shutdown/reboot).'

consolidated_actor_observables_ng

The `consolidated_actor_observables_ng` search table is one of the data sources for the AI Actor Profile dashboard. It checks each Actor association provided in the `consolidated_actor_profiles_ng` and links the corresponding observable information based on the Actor ID. The table is read-only and not subject to modification.



Field	Value Type	Description	Example
actor_id	numeric	ID of the Actor.	190789.0
actor_name	string	Name of the Actor.	HAZY TIGER
confidence	numeric	Risk score from 0 to 100, assigned by ThreatStream's predictive analytics technology to the observable.	70

Field	Value Type	Description	Example
created_ts	date	Time stamp of when the observable was first created in ThreatStream. Date is specified in the following format: `MMM dd, yyyy HH:mm:ss.SSS Z` where • `MMM` - month name • `dd` - day of the month • `yyyy` - year • `HH` - hour (24-clock, 2 digits) • `mm` - minutes • `ss` - seconds • `.SSS` - milliseconds (3 digits) • `Z` - timezone offset from UTC	May 05, 2024 17:15:01.000

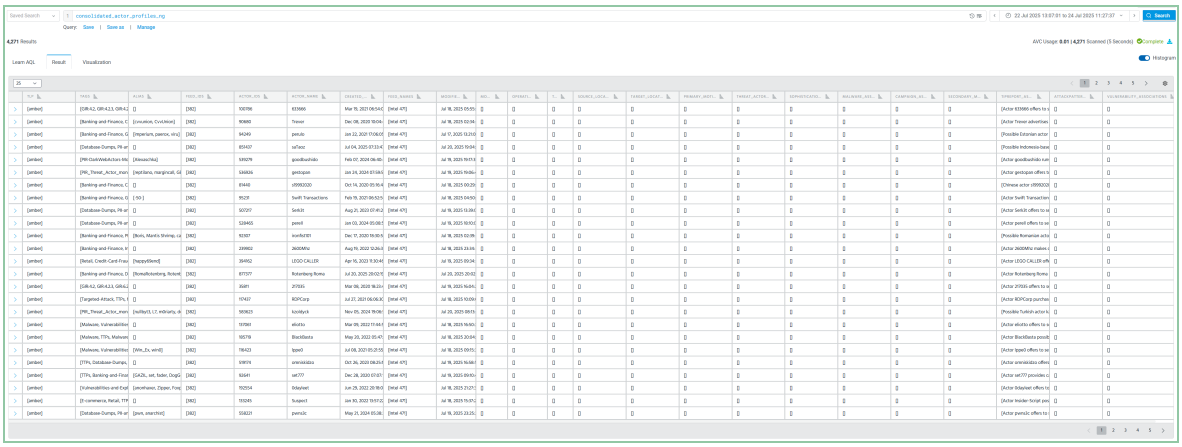
Field	Value Type	Description	Example
feed_name	string	Name of the threat feed that created the observable on ThreatStream.	CrowdStrike Falcon X
indicator	string	Indicator name associated with The Actor.	• folkmusicstreams.com • e462a6710d9b71ecdc41ff7941b2d177 • 194.37.95.173
itype	string	Indicator type of the observable associated with the Actor.	mal_md5

Field	Value Type	Description	Example
modified_ts	date	Time stamp of when the observable was last updated in ThreatStream. Date is specified in the following format: `MMM dd, yyyy HH:mm:ss.SSS Z` where • `MMM` - month name • `dd` - day of the month • `yyyy` - year • `HH` - hour (24-clock, 2 digits) • `mm` - minutes • `ss` - seconds • `.SSS` - milliseconds (3 digits) • `Z` - timezone offset from UTC	Jun 24, 2025 13:40:20.000

Field	Value Type	Description	Example
observable_id	numeric	ID of the observable.	57634912361
severity	string	Criticality associated with the threat feed that supplied the observable Possible values: low, medium, high, very-high.	high

consolidated_actor_profiles_ng

The `consolidated_actor_profiles_ng` search table is one of the data sources for the AI Actor Profile dashboard. It consolidates information about aliases, associations, motivations, and other details about threat actors available on ThreatStream. The table is read-only and not subject to modification.



Field	Value Type	Description	Example
actor_ids	string	ID of the Actor.	784170

Field	Value Type	Description	Example
actor_name	string	Name of the Actor	kasparoff
alias	array of strings	Other names associated with the Actor.	[APT-C-36, Blind Eagle]
attack_pattern_associations	array of strings	Attack Patterns associated with the Actor.	[T1036 - Masquerading, T1566 - Phishing,]
campaign_associations	array of strings	Campaigns associated with the Actor.	[]
created_ts	date	Time stamp of when the Actor was created on ThreatStream. Date is specified in the following format: `MMM dd, yyyy HH:mm:ss.SSS Z` where • `MMM` - month name • `dd` - day of the month • `yyyy` - year • `HH` - hour (24-clock, 2 digits) • `mm` - minutes • `ss` - seconds • `.SSS` - milliseconds (3 digits) • `Z` - timezone offset from UTC	Jun 25, 2025 08:50:46.000 -07

Field	Value Type	Description	Example
feed_ids	numeric array	ID of the feed that created the Actor on ThreatStream.	[383]
feed_names	array of strings	Name of the feed that created the Actor on ThreatStream.	[CrowdStrike Falcon X]
malware_ associations	array of strings	Malware entities associated with the Actor.	[HTRAN, MIMIKATZ,]
modified_ts	date	Time stamp of when the Actor was last updated in ThreatStream. Date is specified in the following format: `MMM dd, yyyy HH:mm:ss.SSS Z` where • `MMM` - month name • `dd` - day of the month • `yyyy` - year • `HH` - hour (24-clock, 2 digits) • `mm` - minutes • `ss` - seconds • `.SSS` - milliseconds (3 digits) • `Z` - timezone offset from UTC	Jun 25, 2025 08:50:57.000 -07

Field	Value Type	Description	Example
motivations	array of strings	Reasons, motivations, or purposes behind the Actor.	[Military, Political]
operation_types	array of strings	Operation types associated with the Actor.	[Cyber Espionage Operations, State Actor / Agency]
primary_ motivation	array of strings	The primary reason, motivation, or purpose behind the Actor. The motivation is why the Actor wishes to achieve the goal (what they are trying to achieve).	[personal-gain]
secondary_ motivations	array of strings	Secondary reasons, motivations, or purposes behind the Actor.	[organizational-gain]
sophistication_ type	array of strings	Level of sophistication observed about the Actor. The Sophistication levels available on ThreatStream follow the Threat Actor Sophistication Vocabulary for STIX. The levels are: No Type (Default), Innovator, Expert, Practitioner, Novice, Aspirant.	[Innovator]
source_locations	array of strings	Source location associated with the Actor (as defined by STIX 2.1).	[Ecuador, Chile]

Field	Value Type	Description	Example
tags	array of strings	Additional comments and context associated with the Actor.	[TTPs, Phishing, Retail, Banking-and-Finance, ABSA_PIR-001_ Phishing]
target_industry	array strings	Target industries associated with the Actor. Target industry options available for selection are defined by the STIX 2.1 Industry Sector vocabulary.	[aerospace, energy, financial-services, government, telecommunications, transportation, utilities]
target_locations	array of strings	Target locations associated with the Actor (as defined by STIX 2.1).	[Belarus]
threat_actor_ types	array of strings	Type of the Actor.	[criminal]
tipreport_ associations	array of strings	Threat Bulletins associated with the Actor.	[Actor SubComandanteVPN offers to sell access to 34 entities worldwide]
tlp	array of strings	Traffic Light Protocol designation for the Actor—red, amber, amber+strict, green, white or clear.	[red]
vulnerability_ associations	array of strings	Vulnerabilities associated with the Actor.	[CVE-2022-30190,]

Appendix B: dgamatch

The dgamatch lookup table is generated by the Anomali Security Analytics domain generated algorithm (DGA) detection feature. The DGA detection feature correlates domain names appearing in event logs with Anomali DGA intelligence.

> **Note:** Correlation features including dgamatch lookups are available with eventlog queries only. See "OCSF Schema Overview" on page 39 for more information.

Field	Value Type	Description	Example
action	string	Action captured by the event.	allowed, blocked
age	number	Event time minus intel first seen.	34
answer	string	Answers for the DNS query.	["10.10.10.10","10.10.10.11"]
app	string	Application detected.	Slack, Google Gmail, LinkedIn
app_type	string	Link type.	splunk, universal_link
asset_dest_criticality	string	Criticality assigned to an asset.	low, medium, high, critical
asset_dest_dept	string	Department of an asset.	engineering, hr, finance

Field	Value Type	Description	Example
asset_dest_hostname	string	Hostname or FQDN of an asset.	qa_jenkins.anomali.com
asset_dest_ip	string	IP address of an asset.	192.168.10.77
asset_dest_location	string	Location of an asset.	Redwood City headquarters
asset_dest_vulnerability	string	Vulnerabilities reported by the asset scanner.	CVE-2022-31214
asset_src_criticality	string	Criticality assigned to an asset.	low, medium, high, critical
asset_src_dept	string	Department of an asset.	engineering, hr, finance
asset_src_hostname	string	Hostname or FQDN of an asset.	qa_jenkins.anomali.com
asset_src_ip	string	IP address of an asset.	192.168.10.77

Field	Value Type	Description	Example
asset_src_location	string	Location of an asset.	Redwood City headquarters
asset_src_vulnerability	string	Vulnerabilities reported by the asset scanner.	CVE-2022-31214
bytes_in	number	Bytes this device or interface received.	347
bytes_out	number	Bytes this device or interface transmitted.	665
category	string	Category of alert (azure security alert).	credentialTheft, ransomware
command_line	string	Complete command string of the spawned process.	C:\Windows\system32\WerFault.exe -pss -s 572 -p 3208 -ip 3208
count	number	Aggregate event count.	4

Field	Value Type	Description	Example
customer_id	string	Reserved for MSSP.	id1
customer_name	string	Reserved for MSSP.	name1
dcid	string	Unique ID of a registered link.	17
dest	string	Destination of network traffic.	52.233.72.222 or www.google.com
dest_ip	string	IP address of the destination.	52.233.72.222
dest_port	number	Destination port of network traffic.	80, 443, 8080
dga_prob	double	DGA probability of a domain.	0.98
dga_tag	string	Additional comments and context.	registered domain

Field	Value Type	Descripti on	Example
ds_id	string	Unique ID of a data source in a registered link. Optional.	5
ds_type	string	Type of a data source in a registered link. Optional.	qradar, sumo or az_sentinel, syslog
event_id	string	A Windows identificati on number that specifies the event type.	4624
event_ name	string	Name of an event.	DnsRequest, ProcessRollup2
event_ time	timesta mp	Time when the event was recorded on the originating device, in UNIX timestamp millisecon ds.	1706309296567

Field	Value Type	Description	Example
file_hash	string	Cryptographic identifier assigned to the file object affected by the event.	ba7816bf8f01cfea414140de5dae2223b00361a 396177a9cb410ff61f20015ad
file_hash2	string	Cryptographic identifier assigned to the file object affected by the event.	56857cfc709d3996f057252c16ec4656f5292802
file_hash3	string	Cryptographic identifier assigned to the file object affected by the event.	e0d123e5f316bef78bfdf5a008837577
file_name	string	Name of the file.	nodepad.exe
file_path	string	Folder containing the file.	C:\Program Files (x86)\VMware\VMware Workstation

Field	Value Type	Description	Example
host	string	Hostname or IP address where link is deployed.	172.18.16.43
http_method	string	HTTP method used in the request.	get, post, delete, patch
http_referrer	string	HTTP referrer used in the request.	https://example.com/
image	string	Executable name of the process.	svchost.exe
indicator	string	Value of an indicator.	123.160.146.123, 2cd3e0df721f0c05e3c385e55e449bf5
ioc_status	string	Status assigned to the indicator.	active, inactive, falsepos
ioc_type	string	Indicator value type.	ip, domain, url, email or md5

Field	Value Type	Description	Example
malware_family	string	A group of applications with similar attack techniques.	conficker
message_id	string	Globally-unique message identifier.	123e4567-e89b-12d3-a456-426655440000
original_file_name	string	Original name of the file that was renamed.	test.pdf
originator	number	Network directionality.	0-inbound, 1-outbound
parent_command_line	string	Complete command string of the parent process.	C:\Windows\System32\svchost.exe -k WerSvcGroup
parent_image	string	Executable name of the parent process.	notepad.exe

Field	Value Type	Descripti on	Example
parent_ process_ id	string	Numeric identifier of the process assigned by the operating system.	4834
platform	string	Operating system of the asset.	win, mac, linux
process_ id	number	Numeric identifier of the process assigned by the operating system.	3683
protocol	string	Network or email protocol.	http, tcp, udp, smtp
query	string	Domain to be resolved.	www.google.com
receiver	string	Email address of the message recipient.	john.doe@amce.org
record_ type	string	DNS resource record type.	A, CNAME, MX, NS, PTR, AAAA

Field	Value Type	Description	Example
reg_key	string	Registry key that the recorded action was applied to.	HKEY_LOCAL_MACHINE\SOFTWARE\RegisteredApplications
reg_value_data	string	Data of the registry value that the recorded action was applied to.	Software\Microsoft\Windows\CurrentVersion\Applets\Wordpad
reg_value_name	string	Name of the registry value that the recorded action was applied to.	Wordpad
return_code	string	HTTP status code or DNS response code.	200, 404, 504, NOERROR, NXDOMAIN
sender	string	Email address of the message sender.	bar@amce.org
sourcetype	string	Device vendor.	PAN Traffic, Cisco ASA, Zscaler

Field	Value Type	Description	Example
src	string	Source of network traffic.	192.168.134.3 or internaltest.qa.amce.org
src_host	string	First non-null value of asset_src_hostname, src, and src_ip.	qa_jenkins.example.com
src_ip	string	IP address of the source.	192.168.34.3
src_port	number	Source port of network traffic.	8080
status	string	Status of the service.	
subject	string	Subject line of the email message.	Your daily Threat Model digest

Field	Value Type	Description	Example
timestamp	timestamp	Time when the event was received by Security Analytics, in UNIX timestamp milliseconds.	1706309296568
title	string	Alert title (azure security alert).	
ttl	number	Time-to-live of the network packet.	3600
url	string	Fully qualified URL of the requested HTTP resource.	https://www.computerhope.com/cgi-bin/search.cgi?q=example%20search
user	string	User that requested the traffic flow.	john.dole@amce.org
user_agent	string	User agent used in the request.	Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:47.0) Gecko/20100101

Appendix C: ignite

The ignite lookup table is generated by the Anomali Security Analytics Ignite detection feature. The Ignite detection feature correlates file hashes appearing in event logs with high fidelity intelligence identifying endpoint filehash IOCs.

Field	Value Type	Description	Example
action	string	Action captured by the event.	allowed, blocked
age	number	Event time minus intel first seen.	34
answer	string	Answers for the DNS query.	["10.10.10.10","10.10.10.11"]
app	string	Application detected.	Slack, Google Gmail, LinkedIn
app_type	string	Link type.	splunk, universal_link
asset_dest_criticality	string	Criticality assigned to an asset.	low, medium, high, critical
asset_dest_dept	string	Department of an asset.	engineering, hr, finance

Field	Value Type	Description	Example
asset_dest_hostname	string	Hostname or FQDN of an asset.	qa_jenkins.example.com
asset_dest_ip	string	IP address of an asset.	192.168.10.77
asset_dest_location	string	Location of an asset.	Redwood city headquarters
asset_dest_vulnerability	string	Vulnerabilities reported by the asset scanner.	CVE-2022-31214
asset_src_criticality	string	Criticality assigned to an asset.	low, medium, high, critical
asset_src_dept	string	Department of an asset.	engineering, hr, finance
asset_src_hostname	string	Hostname or FQDN of an asset.	qa_jenkins.example.com
asset_src_ip	string	IP address of an asset.	192.168.10.77

Field	Value Type	Description	Example
asset_src_location	string	Location of an asset.	Redwood City headquarters
asset_src_vulnerability	string	Vulnerabilities reported by the asset scanner.	CVE-2022-31214
bytes_in	number	Bytes this device or interface received.	347
bytes_out	number	Bytes this device or interface transmitted.	665
category	string	Category of alert (azure security alert).	credentialTheft, ransomware
command_line	string	Complete command string of the spawned process.	C:\Windows\system32\WerFault.exe -pss -s 572 -p 3208 -ip 3208

Field	Value Type	Description	Example
confidence	number	Confidence score assigned to the indicator.	85
count	number	Aggregate event count.	23
customer_id	string	Reserved for MSSP.	id1
customer_name	string	Reserved for MSSP.	name1
dcid	string	Unique ID of a registered link.	17
dest	string	Destination of network traffic.	52.233.72.222 or www.google.com
dest_ip	string	IP address of the destination.	52.233.72.222
dest_port	number	Destination port of network traffic.	80, 443, 8080

Field	Value Type	Description	Example
ds_id	string	Unique ID of a data source in a registered link. Optional.	5
ds_type	string	Type of a data source in a registered link. Optional.	qradar, sumo or az_sentinel, syslog
event_id	string	A Windows identification number that specifies the event type.	4624
event_name	string	Name of an event.	DnsRequest, ProcessRollup2

Field	Value Type	Description	Example
event_time	timestamp	Time when the event was recorded on the originating device, in UNIX timestamp milliseconds.	1705557025030
feed_id	number	Source feed ID in ThreatStream of the indicator.	345
feed_name	string	Source of the indicator.	crowdstrike, recorded future
file_hash	string	Cryptographic identifier assigned to the file object affected by the event.	e0d123e5f316bef78bfdf5a008837577

Field	Value Type	Description	Example
file_hash2	string	Cryptographic identifier assigned to the file object affected by the event.	56857cfc709d3996f057252c16ec4656f5292802
file_hash3	string	Cryptographic identifier assigned to the file object affected by the event.	ba7816bf8f01cfea414140de5dae2223b00361a396177a9cb410ff61f20015ad
file_name	string	Name of the file.	nodepad.exe
file_path	string	Folder containing the file.	C:\Program Files (x86)\VMware\VMware Workstation
host	string	Hostname or IP address where link is deployed.	192.0.2.1

Field	Value Type	Description	Example
http_method	string	HTTP method used in the request.	get, post, delete, patch
http_referrer	string	HTTP referrer used in the request.	https://example.com/
id	number	ID of an indicator.	34566
ignite_match_id	string	Unique ID.	
image	string	Executable name of the process.	svchost.exe
indicator	string	Value of an indicator.	123.160.146.123, 2cd3e0df721f0c05e3c385e55e449bf5
ioc_status	string	Status assigned to the indicator.	active, inactive, falsepos
ioc_type	string	Indicator value type.	ip, domain, url, email or md5
itype	string	Indicator type.	apt_ip

Field	Value Type	Description	Example
message_id	string	Globally-unique message identifier.	123e4567-e89b-12d3-a456-426655440000
original_file_name	string	Original name of the file that was renamed.	test.pdf
parent_command_line	string	Complete command string of the parent process.	C:\Windows\System32\svchost.exe -k WerSvcGroup
parent_image	string	Executable name of the parent process.	notepad.exe
parent_process_id	string	Numeric identifier of the process assigned by the operating system.	4834
platform	string	Operating system of the asset.	win, mac, linux

Field	Value Type	Description	Example
process_id	number	Numeric identifier of the process assigned by the operating system.	3683
protocol	string	Network or email protocol.	http, tcp, udp, smtp
query	string	Domain to be resolved.	www.google.com
receiver	string	Email address of the message recipient.	john.doe@amce.org
record_type	string	DNS resource record type.	A, CNAME, MX, NS, PTR, AAAA
reg_key	string	Registry key that the recorded action was applied to.	HKEY_LOCAL_MACHINE\SOFTWARE\RegisteredApplications

Field	Value Type	Description	Example
reg_value_data	string	Data of the registry value that the recorded action was applied to.	Software\Microsoft\Windows\CurrentVersion\Applets\Wordpad
reg_value_name	string	Name of the registry value that the recorded action was applied to.	Wordpad
return_code	string	HTTP status code or DNS response code.	200, 404, 504, NOERROR, NXDOMAIN
sender	string	Email address of the message sender.	bar@amce.org
severity	string	Severity assigned to the indicator.	low, medium, high, very-high

Field	Value Type	Description	Example
sourcetype	string	Device vendor.	PAN Traffic, Cisco ASA, Zscaler
src	string	Source of network traffic.	192.168.134.3 or internaltest.qa.amce.org
src_host	string	First non-null value of asset_src_hostname, src, and src_ip.	qa_jenkins.example.com
src_ip	string	IP address of the source.	192.168.34.3
src_port	number	Source port of network traffic.	8080
status	string	Status of the service.	
subject	string	Subject line of the email message.	Document shared with you

Field	Value Type	Description	Example
tags	string	Additional comments and context.	popularity=low, malware
timestamp	timestamp	Time when the event was received by Security Analytics, in UNIX timestamp milliseconds.	1705557025030
title	string	Alert title (azure security alert).	
ttl	number	Time-to-live of the network packet.	3600
url	string	Fully qualified URL of the requested HTTP resource.	https://www.computerhope.com/cgi-bin/search.cgi?q=example%20search

Field	Value Type	Description	Example
user	string	User that requested the traffic flow.	john.dole@amce.org
user_agent	string	User agent used in the request.	Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:47.0) Gecko/20100101

Appendix D: iocmatch

The iocmatch lookup table is generated by the Anomali Security Analytics IOC correlation engine. The IOC correlation engine correlates your event logs with your ThreatStream IOCs.

> **Note:** Correlation features including iocmatch lookups are available with eventlog queries only. See "OCSF Schema Overview" on page 39 for more information.

Name	Value Type	Description	Example
action	string	Action captured by the event.	allowed, blocked
actor_ids	list	Associated actor IDs in ThreatStream.	[2342, 2313]
age	number	Event time minus intel first seen.	34
answer	string	Answers for the DNS query.	["10.10.10.10","10.10.10.11"]
app	string	Application detected.	Slack, Google Gmail, LinkedIn
app_type	string	Link type.	splunk, universal_link

Name	Value Type	Description	Example
asn	string	Autonomous System (AS) number.	48780
asset_dest_criticality	string	Criticality assigned to an asset.	low, medium, high, critical
asset_dest_dept	string	Department of an asset.	engineering, hr, finance
asset_dest_hostname	string	Hostname or FQDN of an asset.	qa_jenkins.example.com
asset_dest_ip	string	IP address of an asset.	192.168.10.77
asset_dest_location	string	Location of an asset.	Redwood city headquarters
asset_dest_vulnerability	string	Vulnerabilities reported by the asset scanner.	CVE-2022-31214
asset_src_criticality	string	Criticality assigned to an asset.	low, medium, high, critical

Name	Value Type	Description	Example
asset_src_dept	string	Department of an asset.	engineering, hr, finance
asset_src_hostname	string	Hostname or FQDN of an asset.	qa_jenkins.example.com
asset_src_ip	string	IP address of an asset.	192.168.10.77
asset_src_location	string	Location of an asset.	Redwood City headquarters
asset_src_vulnerability	string	Vulnerabilities reported by the asset scanner.	CVE-2022-31214
attackpattern_ids	list	Associated attack pattern IDs in ThreatStream.	[2342, 2313]
bytes_in	number	Bytes this device or interface received.	347

Name	Value Type	Description	Example
bytes_out	number	Bytes this device or interface transmitted.	665
campaign_ids	list	Associated campaign IDs in ThreatStream.	[2342, 2313]
category	string	Category of alert (azure security alert).	credentialTheft, ransomware
classification	string	Classification of an intel.	public or private
command_line	string	Complete command string of the spawned process.	C:\Windows\system32\WerFault.exe -pss -s 572 -p 3208 -ip 3208
confidence	number	Confidence score assigned to the indicator.	85

Name	Value Type	Description	Example
count	number	Aggregate event count.	23
country	string	Country associated with the indicator.	US
created_ts	timestamp	First seen on ThreatStream.	1705557025030
customer_id	string	Reserved for MSSP.	id1
customer_name	string	Reserved for MSSP.	name1
dcid	string	Unique ID of a registered link.	17
dest	string	Destination of network traffic.	52.233.72.222 or www.google.com
dest_ip	string	IP address of the destination.	52.233.72.222

Name	Value Type	Description	Example
dest_port	number	Destination port of network traffic.	80, 443, 8080
ds_id	string	Unique ID of a data source in a registered link. Optional.	5
ds_type	string	Type of a data source in a registered link. Optional.	qradar, sumo or az_sentinel, syslog
event_id	string	A Windows identification number that specifies the event type.	4624
event_name	string	Name of an event.	DnsRequest, ProcessRollup2

Name	Value Type	Description	Example
event_time	timestamp	Time when the event was recorded on the originating device, in UNIX timestamp milliseconds.	1705557025030
feed_id	number	Source feed ID in ThreatStream of the indicator.	345
feed_name	string	Source of the indicator.	crowdstrike, recorded future
file_hash	string	Cryptographic identifier assigned to the file object affected by the event.	e0d123e5f316bef78bfdf5a008837577

Name	Value Type	Description	Example
file_hash2	string	Cryptographic identifier assigned to the file object affected by the event.	56857cfc709d3996f057252c16ec4656f5292802
file_hash3	string	Cryptographic identifier assigned to the file object affected by the event.	ba7816bf8f01cfea414140de5dae2223b00361a396177a9cb410ff61f20015ad
file_name	string	Name of the file.	nodepad.exe
file_path	string	Folder containing the file.	C:\Program Files (x86)\VMware\VMware Workstation
host	string	Hostname or IP address where link is deployed.	192.0.2.1

Name	Value Type	Description	Example
http_method	string	HTTP method used in the request.	get, post, delete, patch
http_referrer	string	HTTP referrer used in the request.	https://example.com/
id	number	ID of an indicator.	34566
image	string	Executable name of the process.	svchost.exe
import_session_id	number	ID of import session in which the indicator was imported.	234
indicator	string	Value of an indicator.	123.160.146.123, 2cd3e0df721f0c05e3c385e55e449bf5

Name	Value Type	Description	Example
ioc_match_id	string	Unique ID, having format {orgid}_{event_id}_0_{ioc_id}.	1773_79618005142_0_59324653376
ioc_status	string	Status assigned to the indicator.	active, inactive, falsepos
ioc_type	string	Indicator value type.	ip, domain, url, email or md5
itype	string	Indicator type.	apt_ip
latitude	double	Latitude associated with the Geo location of the IP address.	37.579
longitude	double	Longitude associated with the Geo location of the IP address.	126.975

Name	Value Type	Description	Example
message_id	string	Globally-unique message identifier.	123e4567-e89b-12d3-a456-426655440000
modified_ts	timestamp	Last updated on ThreatStream.	1705557025030
org	string	Registered owner (organization) of the IP address associated with the indicator.	Acme Corp
original_file_name	string	Original name of the file that was renamed.	test.pdf
originator	number	Network directionality.	0-inbound, 1-outbound
parent_command_line	string	Complete command string of the parent process.	C:\Windows\System32\svchost.exe -k WerSvcGroup

Name	Value Type	Description	Example
parent_image	string	Executable name of the parent process.	notepad.exe
parent_process_id	string	Numeric identifier of the process assigned by the operating system.	4834
platform	string	Operating system of the asset.	win, mac, linux
process_id	number	Numeric identifier of the process assigned by the operating system.	3683
protocol	string	Network or email protocol.	http, tcp, udp, smtp
query	string	Domain to be resolved.	www.google.com

Name	Value Type	Description	Example
receiver	string	Email address of the message recipient.	john.doe@amce.org
record_type	string	DNS resource record type.	A, CNAME, MX, NS, PTR, AAAA
reg_key	string	Registry key that the recorded action was applied to.	HKEY_LOCAL_MACHINE\SOFTWARE\RegisteredApplications
reg_value_data	string	Data of the registry value that the recorded action was applied to.	Software\Microsoft\Windows\CurrentVersion\Applets\Wordpad

Name	Value Type	Description	Example
reg_value_name	string	Name of the registry value that the recorded action was applied to.	Wordpad
resource_uri	string	URI for the indicator	
return_code	string	HTTP status code or DNS response code.	200, 404, 504, NOERROR, NXDOMAIN
sender	string	Email address of the message sender.	bar@amce.org
severity	string	Severity assigned to the indicator.	low, medium, high, very-high
sourcetype	string	Device vendor.	PAN Traffic, Cisco ASA, Zscaler
src	string	Source of network traffic.	192.168.134.3 or internaltest.qa.amce.org

Name	Value Type	Description	Example
src_host	string	First non-null value of asset_src_hostname, src, and src_ip.	qa_jenkins.example.com
src_ip	string	IP address of the source.	192.168.34.3
src_port	number	Source port of network traffic.	8080
status	string	Status of the service.	
subject	string	Subject line of the email message.	Document shared with you
tags	string	Additional comments and context.	popularity=low, malware

Name	Value Type	Description	Example
timestamp	timestamp	Time when the event was received by Security Analytics, in UNIX timestamp milliseconds.	1705557025030
tipreport_ids	list	Associated threat bulletin IDs in ThreatStream.	[2342, 2313]
title	string	Alert title (azure security alert).	
trusted_circle_ids	list	List of trusted circle ids assigned in ThreatStream to the indicator.	[1, 45]

Name	Value Type	Description	Example
trusted_circle_names	list	List of trusted circle names.	["circle1", "circle45"]
ttl	number	Time-to-live of the network packet.	3600
ttp_ids	list	Associated TTP IDs in ThreatStream.	[5473, 34724]
url	string	Fully qualified URL of the requested HTTP resource.	https://www.computerhope.com/cgi-bin/search.cgi?q=example%20search
user	string	User that requested the traffic flow.	john.dole@amce.org
user_agent	string	User agent used in the request.	Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:47.0) Gecko/20100101

Name	Value Type	Description	Example
vulnerability_ids	list	Associated vulnerability IDs in ThreatStream.	[5473, 34724]

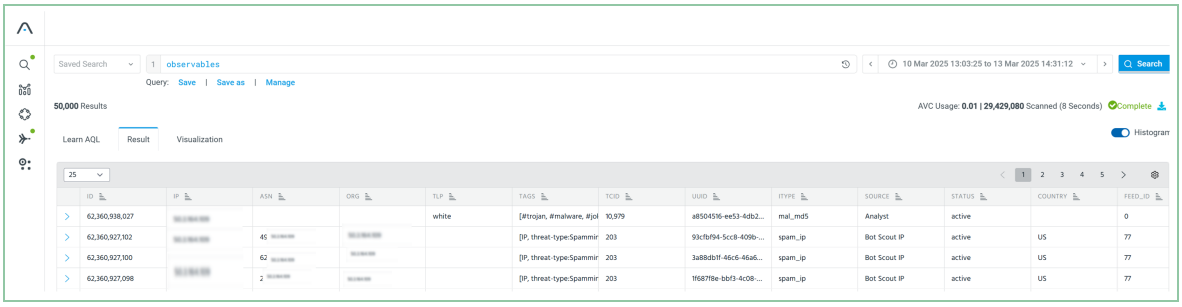
Appendix D: observables

AQL Search supports the out-of-the box `observables` lookup table that contains ThreatStream observable data from the last 30 days.

To view or search for these ThreatStream observables, enter the following lookup table name into the AQL Search box: `observables`.

You can also access the `observables` table on the Lookup Tables page. See "Using Lookup Tables" on page 125 for more information.

Observable data is refreshed once daily.



Field	Value Type	Description	Example
asn	number	The Autonomous System Number for the IP associated with the observable.	37,963

Field	Value Type	Description	Example
confidence	number	Indicates the ThreatStream-Assigned confidence of an observable being malicious. See Observable Confidence in ThreatStream for more information.	75
country	string	Two-letter ISO country code for the IP associated with the observable.	• US • CN • DE
created_by	string	Email address of the user who submitted the import job containing the observable.	
created_ts	date	UTC time stamp of when the observable was first created in ThreatStream.	
expiration_ts	date	Time stamp of when intelligence will expire on ThreatStream, in UTC format.	2017-01-26T00:00:00.
extended_ source			
feed_id	numeric	ID of the threat feed that created the observable on ThreatStream.	
id		Unique ID of a registered link.	17
ip	string	IP associated with the observable.	52.233.72.222

Field	Value Type	Description	Example
is_public		Specific nation or geographical region to which data, materials, or information is being directed or delivered.	us
import_ session_id	numeric	ID of the import session that created the observable on ThreatStream API.	
import_source	string	Original source of the observable. Values are only displayed for observables manually imported through the ThreatStream API user interface by your organization. Import source for observables owned by other organizations is not visible.	
ioc_type	string	Type of the observable.	• ip • domain • email • string • md5 • ipv6 • url • phone_num
ioc_type_v2	string	Unified hash label for all hash-based observables.	hash

Field	Value Type	Description	Example
ip_network_start			
ip_network_end			
itype	string	ThreatStream indicator type.	• c2_ip • compromised_email • apt_md5 • mal_md5
itype_v2	string	Unified hash label for ThreatStream indicator types.	mal_hash
latitude	numeric	Latitude associated with the Geo location of the IP.	12.975
longitude	numeric	Longitude associated with the Geo location of the IP.	• -80.193 • 77.591
meta			
modified_ts	date	UTC time stamp of when the observable was last updated in ThreatStream.	2014-10-02T20:44:35.
org	string	ThreatStream organization that owns the observable.	T-Mobile USA
owner_organization_id	numeric	ID of the organization owner.	2

Field	Value Type	Description	Example
rdns	string	Domain name (obtained through reverse domain name lookup) associated with the IP address that is associated with the observable.	
retina_ confidence	numeric	Confidence score calculated using ThreatStream's machine learning-based scoring engine. This score is only calculated for IP-, domain-, and URL-based observables. When reetina score is –1, it implies that a valid score could not be calculated or the source reported confidence was exclusively used.	67
source_created	date	Time stamp of when the entity was created by its original source.	
source_ modified	date	Time stamp of when the entity was last updated by its original source.	
source_ reported_ confidence	numeric	Indicates the source reported confidence of an observable being malicious. See Observable Confidence in ThreatStream for more information.	82

Field	Value Type	Description	Example
status	string	Current state of the observable in ThreatStream.	• active • inactive • falsepos
subtype	string	For hash observables—those with type=md5—subtype provides additional metadata on the type of hash associated with the observable.	• md5 • sha1 • sha256 • sha512
tags	string	Additional comments and context associated with the observable when it was imported from its original threat feed.	
threat_type	string	Summarized threat type of the observable.	• malware • compromised • apt • c2
trusted_circle_ids	numeric	ID of the trusted circle with which the observable is shared.	
type	string	Data type of the observable.	• ip • domain • email • md5 • url

Field	Value Type	Description	Example
updated_id		Event message available when the "store raw logs" option has been configured.	1.0 2017-12-13T08:16:02.130Z Z123412341234 example.com A NOERROR UDP Region 192.168.1.1 -

Appendix E: tmmatch

The tmmatch lookup table is generated by the Anomali Security Analytics threat model correlation feature. The threat model correlation feature produces a summary of threat model matches found in your event logs.

> **Note:** Correlation features including tmmatch lookups are available with eventlog queries only. See "OCSF Schema Overview" on page 39 for more information.

Name	Value Type	Description	Example
event_time	timestamp	Time when the event was recorded on the originating device, in UNIX timestamp milliseconds.	1706309472445
ioc_match_id	string	Unique ID, having format {orgid}_{event_id}_0_{ioc_id}.	1773_79618005142_0_59324653376
timestamp	timestamp	Time when the event was received by Security Analytics, in UNIX timestamp milliseconds.	1706309472445
tm_id	number	Threat model ID in ThreatStream.	1459204
tm_match_id	string	Unique ID, having format {orgid}_{event_id}_0_{ioc_id}_{tm_id}.	1773_427842772067_3_60760916306_1459204
tm_name	string	Threat model name.	T1071 - Application Layer Protocol
tm_type	string	Threat model type.	actor, attackpattern, campaign, tb, ttp or vulnerability

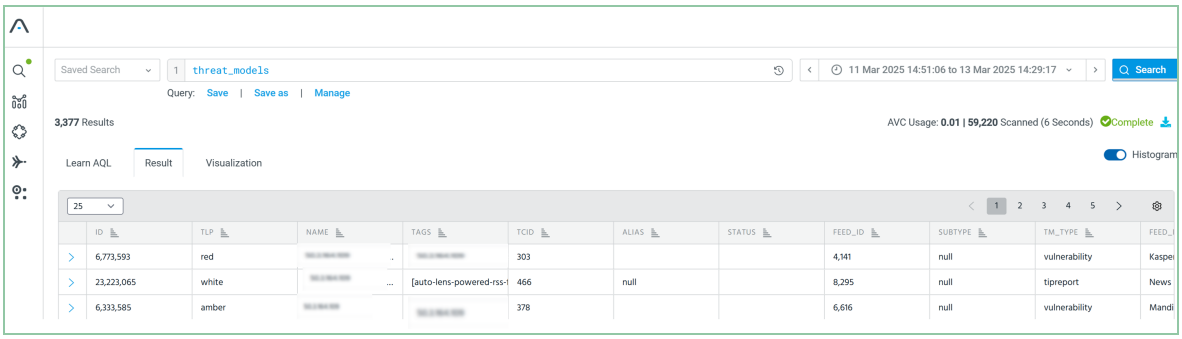
Appendix E: threat_models

AQL Search supports the out-of-the box `threat_models` lookup table that contains ThreatStream threat model data from the last 90 days.

To search for these threat models, enter the following lookup table name into the AQL Search box: `threat_models`.

You can also access the `threat_models` table on the Lookup Tables page. See "Using Lookup Tables" on page 125 for more information.

Threat model data is refreshed once daily.



Field	Value Type	Description	Example
alias	string	Other names by which the entity is known (Actors, Attack Patterns, Campaigns, Infrastructure, Intrusion Sets, Malware, Tools, TTPs, Vulnerabilities only).	user -> UserName
assignee_user_id	double	ID of the user assigned to the threat model.	393
c2_port	array of strings	Network port(s) used for C2 communications. Applicable to Malware only.	[8080, 443]
c2_protocol	array of strings	Protocol(s) used for C2 traffic. Applicable to Malware only.	[HTTP, HTTPS, DNS]

Field	Value Type	Description	Example
capabilities	array of strings	The functions and features of the malware.	[Keylogging, Credential theft, File exfiltration]
created_ts	bigint	Time stamp of when the entity was created on ThreatStream, in UTC format. Date can be specified as follows: • In this format: YYYY-MM-DDThh:mm:ss, where T denotes the start of the value for time. For example, 2014-10-02T20:44:35. T must be capitalized. • As a relative time unit, in this format: -<n><unit>, where n is a whole number and unit is w, d, h, m, s (for week, days, hour, minutes, and seconds, respectively). For example, -2w denotes two weeks, starting NOW. Units must be lowercase.	2014-10-02T20:44:35

epss_percentile	double	EPSS percentile.
		When EPSS data is unavailable the field value is `null`.
		The field works with `sort` and `where` clauses and supports the following operators: `>`, `<`, `>=`, and `<=`.
		For example:
		`threat_models`
		`| where tm_type == "vulnerability"`
		`| sort epss_percentile desc`
			• 0.894
			• 1
			• null

Field	Value Type	Description	Example
epss_score	double	EPSS score. When EPSS data is unavailable the field value is `null`. The field works with `sort` and `where` clauses and supports the following operators: `>`, `<`, `>=`, and `<=`. For example: `threat_models	where tm_type == "vulnerability" and epss_score > 0.5`	• 0.944 • null • 0.754
cvss2_score	double	CVSS 2.0 Score (Vulnerability entities only). Scores range from 0-10.	• 5 • null
cvss3_score	double	CVSS 3.0 Score (Vulnerability entities only). Scores range from 0-10.	• 9.8
end_date	bigint	When the campaign stopped (if applicable).	2024-12-15
event_time	bigint	UTC time stamp of when the observable was first created in ThreatStream.	Jun 20, 2025 11:53:15.000 -07
execution_platforms	array of strings	OS or environments the malware runs on.	Windows, Linux, macOS, Android
feed_id	bigint	Numeric ID of the threat feed that provided the Threat Model entity.	2017-01-26T00:00:00.
feed_name	string	Name of the feed from which the entity originates.	MITRE ATT&CK PATTERN
first_seen	bigint		

Field	Value Type	Description	Example
id	bigint	ID of the attack pattern entity. This ID is automatically assigned to the entity by ThreatStream at the time of creation. You can use this ID to associate the attack pattern entity with other Threat Model entities.	17
is_family	boolean	Indicates if this entry describes a malware family.	true, false
is_mitre	boolean	Determines whethere the entries are directly sourced from MITRE ATT&CK	true, false
locations	array of strings	Specific nation or geographical region to which data, materials, or information is being directed or delivered.	us
modified_ts	bigint	UTC time stamp of when the observable was last updated in ThreatStream. Date can be specified as follows: • In this format: YYYY-MM-DDThh:mm:ss, where T denotes the start of the value for time. For example, 2014-10-02T20:44:35. T must be capitalized. • As a relative time unit, in this format: -<n><unit>, where n is a whole number and unit is w, d, h, m, s (for week, days, hour, minutes, and seconds, respectively). For example, -2w denotes two weeks, starting NOW. Units must be lowercase.	Jun 20, 2025 11:53:15.000 -07
mitre_data_sources	array of strings	Types of telemetry or logs that can be used to detect the technique.	[Malware reverse engineering, Process monitoring]

Field	Value Type	Description	Example
mitre_defense_bypassed	array of strings	Identifies the defensive controls that the technique is known to circumvent.	[Signature-based detection]

Field	Value Type	Description	Example
mitre_ detection	string	Ways in which an org detects the use of a technique. This usually contains hints on the relevant behaviors, artifacts, or anomalies.	Depending on the method used to pad files, a file-based signature may be capable of detecting padding using a scanning or on-access based tool. When executed, the resulting process from padded files may also exhibit other behavior characteristic s of being used to conduct an intrusion such as system and network information Discovery or Lateral Movement, which could be used as event indicators

Field	Value Type	Description	Example
			that point to the source file.
mitre_name	string	Name of the ATT&CK technique or sub-technique, including its ID.	T1027.001 - Obfuscated Files or Information: Binary Padding
mitre_id	string	The MITRE ATT&CK technique or sub-technique identifier.	T1027.001
mitre_parent_name	string	If the entry is a sub-technique, this field stores the parent technique's name.	T1027 - Obfuscated Files or Information
mitre_parent_id	string	The MITRE ATT&CK ID of the parent technique for a sub-technique.	T1027
mitre_parent_attackpattern	integer	An internal or database-specific numeric ID representing the parent technique in the dataset.	9591
mitre_tactic	array of strings	Signifies the higher-level adversary objective.	defense-evasion
mitre_type	string	Type of the MITRE ATT&CK.	enterprise-attack

Field	Value Type	Description	Example
name	string	Name of the entity.	T1027.001 - Obfuscated Files or Information: Binary Padding
owner_user_id	bigint	ID of the ThreatStream user that created the entity.	T-Mobile USA
organization_id	bigint	ID of the organization that owns the entity.	2
primary_motivation	string	Primary reason, motivation, or purpose behind the Actor. The motivation is why the Actor wishes to achieve the goal. This field is relevant only to Actors and Intrusion Sets.	
product_name	string	Identifies which product(s) from the listed vendors are impacted.	Exchange Server 2016
publication_status	string	Current stage in the publishing workflow.	• new • review_requested • reviewed • published
secondary_motivation	array of strings	The additional, secondary goals of a threat actor. This field is relevant only to Actors and Intrusion Sets.	Destruction

Field	Value Type	Description	Example
sophistication	string	Assessment of skill, resources, and operational maturity. This field is relevant only to Actors.	No Type (Default), Novice, Innovator, Expert, Practitioner, Aspirant.
source_locations	array of strings	Source locations associated with the Threat Model entity. Possible values include letter codes of regions, countries, and administrative areas (US States and Washington DC only) as defined by STIX 2.1.	Here are some examples: • `easternafrica` • `central-asia` • `europe` • africa • `NL` • `GB` • `US-FL` • `US-MO`
start_date	bigint	Time when a campaign was known to have started. Date (in UTC) must be specified in this format: YYYYMMDDThhmmss, where T denotes the start of the value for time.	
status	string	Current state of the observable in ThreatStream.	• ongoing • historic • future

Field	Value Type	Description	Example
subtype	array of strings	Provides further categorization of threat model information, as per the following threat model types: • Actor • Infrastructure • Malware • Signature • Tool	• **Actor** activist, competitor, crime-syndicate, criminal, hacker, insider-accidental, insider-disgruntled, nation-state, sensationalist, spy, terrorist, unknown • **Infrastructure** amplification, anonymization, botnet, command-and-control, exfiltration, hosting-malware, hosting-target-lists, phishing, reconnaissance,

Field	Value Type	Description	Example
			staging, undefined • **Malware** adware, backdoor, bot, bootkit, ddos, downloader, dropper, exploit-kit, keylogger, ransomware, remote-access-trojan, resource-exploitation, rogue-security-software, rootkit, screen-capture, spyware, trojan, unknown, virus, webshell, wiper, worm • **Signature** Snort,

Field	Value Type	Description	Example
			YARA, CybOX, OpenIOC, ClamAV, Suricata, Bro, Carbon Black Query, Custom, Splunk Query, RSA NetWitness, Sigma • **Tool** denial-of-service, exploitation, information-gathering, network-capture, credential-exploitation, remote-access, vulnerability-scanning, unknown

Field	Value Type	Description	Example
tags	array of strings	Additional comments and context associated with the observable when it was imported from its original threat feed.	[T1556.009, attack-pattern--ceaeb6d8-95ee-4da2-9d42-dc6aa6ca43ae]
target_industry	array of strings	Target industries associated with the Threat Model entity. Possible values are defined by the STIX 2.1 Industry Sector vocabulary	• chemical • financial-services • government-regional
tcid	integer	Unique reference of a MITRE ATT&CK technique, sub-technique, or related object.	145
threat_actor_types	array of strings	Categorizes the type of Actor based on operational profile.	Nation-state
tlp	string	TLP color associated with the Threat Model Entity. The TLP color provides a mechanism to communicate to consumers of the information whether further dissemination of this information is allowed; if yes, how freely can this information be distributed.	• white • clear • green • red • amber+strict
tm_type	string	Type of the threat model.	tipreport vulnerability

Field	Value Type	Description	Example
trusted_circle_ids	array of integers	ID of the trusted circle with which the observable is shared. These IDs can be used for querying entities associated with specified trusted circles.	[145]
trusted_circle_names	array of integers	Names of trusted circles.	[Anomali Threat Research Premium]
vendors	string	Vendor(s) affected by the Vulnerability.	[Microsoft, Cisco]

Appendix F: Reference: Date Time Format Code

The following table describes the date-time format codes used with AQL `calc` operator timestamp conversion functions.

Format Code	Description	Example
`%Y`	Four-digit year	2025
`%y`	Two-digit year	25
`%m`	Two-digit month (01-12)	01
`%b`	Name of the month	Jan
`%d`	Two-digit day of the month (01-31)	05
`%H`	Two-digit hour (00-23)	13
`%h`	Two-digit hour (01-12)	05
`%M`	Two-digit minute (00-59)	37
`%S`	Two-digit second (00-59)	45
`%Q`	Three-digit milliseconds (000-999)	300

Note: If the string format is incorrect or the date is invalid, the AQL query returns an error message similar to the following:

```
INVALID_FUNCTION_ARGUMENT: Invalid format
```

Using the Search RESTful API

AQL Search is accessible through a REST API. The API allows you to create a new search, monitor the status of the search job, and successfully retrieve the search results.

> **Notes:**
>
> - Only the API endpoints described in this document are supported.
> - Only 10 Create New Search API requests per minute per organization are allowed.

About the API

The API offers bi-directional interaction with Search.

Accessing the API

To access the API related to Search, use the following base URL:

```
https://api.threatstream.com/api/v1/xdr/<resource>/
```

where

`<resource>` is an API endpoint; for example, `/search/jobs`

Authenticating to Search

Making requests through the API requires authenticating to ThreatStream using your username (the email address associated with your ThreatStream account) and your dedicated API Key. You can find your username and API Key on the My Profile tab within ThreatStream settings.

Specifying your username and API Key in the header of the request is the most secure method of authentication. Anomali recommends using this authentication method.

```
curl 'https://api.threatstream.com/api/v1/xdr/<resource>/' -H
'Authorization: apikey <username>:<api_key>'
```

Once you authenticate, each API call uses your API Key to successfully make API requests.

Create a New Search

To trigger a new search and create a new search job.

> **Note:**
> Only 10 new search API requests per minute per organization are allowed.

Request

`/search/jobs/`

HTTP method: POST

Attributes: See table below.

Attributes

Attribute	Type	Description
query	string	Required. Any search query. For information on how to create a query, see "Understanding the Search Bar" on page 22.
source	string	(Optional) source of the request, set as `third_party_<your_app_name>`. For example, `source="third_party_mynewapp"`. **Default**: `source` is set to "`third_party`", if you don't provide a value for `your_app_name`.

time_range	string	Required. Time range (in UNIX timestamp milliseconds) and the timezone in which records are searched. For example, you can issue a search in the records created within the last 15 minutes. For example:
		`"time_range": {`
		`"from": 1679777426000,`
		`"to": 1680123026000,`
		`"timezone":"America/Los_Angeles"`
		`}`

Response

Format: JSON

Attributes: See table below.

Attributes

Attribute	Type	Description
job_id	string	Job ID of the search job.

Examples

Request:

curl example:

```
curl -X 'POST' 'https://api.threatstream.com/api/v1/xdr/search/jobs/' \
    -H 'Authorization: apikey <username>:<api_key>' \
    -H 'accept: application/json' -H 'Content-Type: application/json' \
    -d '{"query": 'eventlog | where sourcetype contains \"crowdstrike\" or
    sourcetype contains \"edr\" \
    | aggr count as cnt by src | sort cnt desc | limit 100','source": "third_party_
    mynewapp",\
    "time_range": {"from": 1679777426000,"to": 1680123026000,"timezone":
    "America/Los_Angeles"}}'
```

Python example:

```
import requests
import json

response = requests.post(
        'https://api.threatstream.com/api/v1/xdr/search/jobs/',
        headers={
                'Accept': 'application/json',
                'Content-Type': 'application/json',
                'Authorization': 'apikey <username>:<api_key>'
        },    data=json.dumps({
                "source": "third_party_mynewapp",
                "query": 'eventlog | where sourcetype contains "crowd-strike" or \
                                        sourcetype contains "edr" | \
                                        aggr count as cnt by src | \
                                        sort cnt desc | limit 100',
                "time_range": {
                        "from": 1679777426000,
                        "to": 1680123026000,
                        "timezone": "America/Los_Angeles"
                }
        })
)

if response.status_code == 200:
        try:
                response_data = response.json()
                job_id = response_data.get('job_id')
```

```
                print("Job ID: ",job_id)
        except ValueError:
                print("Response is not in JSON format:",response.text)
else:
        print("Error:", response.status_code, response.text)
```

Response:

```
{
        "job_id": 6f0f5c333a8b48e1b2659b4578c1b51f
}
```

Successful response status code: 200

Failed response error codes: 400, 500

Get Search Status

To get the status of a search using its job ID.

Request

`/search/jobs/{job_id}/`

HTTP method: GET

Attributes: See table below.

Attributes

Attribute	Type	Description
job_id	string	Required. Job ID of the search job.

Response

Returns a JSON object that contains the following information:

- Histogram information
- Aggregation status
- Progress status
- Scan count
- Job status

- QUEUED
- CANCELLED
- RUNNING
- DONE
- FAILED

Format: JSON

Attributes: See table below.

Attributes

Attribute	Type	Description
bucket_length	integer	Width of each histogram bin depending on the search duration.
end_time	integer	End timestamp of the time interval of the search query, in UNIX timestamp milliseconds. For example, if the search query interval lies between 0-100, then `start_time`=0 and `end_time`=100.
histogram_buckets	list	Number of histogram buckets used to bucket the search result. Attributes: • `count`—Number of search results that matched the search query in a particular 10 minute histogram bin. • `length`—Width of each histogram bin depending on the search duratoin. • `start_time`—Timestamp of a particular histogram bin.

Attribute	Type	Description

is_ aggregated	boolean	Whether the search results are aggregated.
		TRUE—Query is an aggregation operation. For example:
		`eventlog | where sourcetype contains \"crowdstrike\" or sourcetype contains \"edr\" | aggr count as cnt by src | sort cnt desc | limit 100"`
		FALSE—Query is not an aggregation operation. For example:
		`eventlog | fields src_ip, dest_ip`

num_of_ bucket	integer	Number of buckets in which you receive all the search results. Default is 60.
progress	numeric	Indicates the search progress. Numeric float value between 0 and 1.
scanned	integer	Number of records scanned by the search query.
start_time	integer	Start timestamp of the time interval of the search query, in UNIX timestamp milliseconds. For example, if the search query interval lies between 0-100, then `start_time`=0 and `end_time`=100.

Attribute	Type	Description
status	string	Status of the search. Possible options: • `QUEUED`: Search job is waiting for execution. • `CANCELLED`: Query was canceled while it was running. • `RUNNING`: Search in progress. • `DONE`: Search query is complete and the search results are retrieved. • `FAILED`: When no result is returned due to many reasons, such as "Timeout", "Server error", "Job Id not found", or "Unexpected error". **Note:** If more than 120 seconds lapse in the `QUEUED` status, you will encounter a `FAILED` status. You can reissue a new search.
total	integer	Total number of records matched by the search query.

Examples

Request:

curl example:

```
curl -X 'GET'
    'https://api.threatstream.com/api/v1/xdr/search/jobs/6f5f5c716a8b48e1b2659b5918c
    1b51f/' \
    -H 'Authorization: apikey <username>:<api_key>' \
    -H 'accept: application/json'
```

Python example:

```
import requests
response = requests.get(

"https://api.threatstream.com/api/v1/xdr/search/jobs/6f5f5c716a8b48e1b2659b5918c1b51
f/",
       headers={
```

```
                'Accept': 'application/json',
                'Authorization': 'apikey <username>:<api_key>'
        }
)
```

Response:


```
{
        "bucket_length": 1200000,
        "end_time": 1727650800000,
        "histogram_buckets": [
                {
                        "count": 0,
                        "length": 1200000,
                        "start_time": 1727613600000
                },
                {
                        "count": 0,
                        "length": 1200000,
                        "start_time": 1727614800000
                }
        ],
        "is_aggregated": false,
        "num_of_bucket": 60,
        "progress": 1.0,
        "scanned": 35004,
        "start_time": 1727613600000,
        "status": "DONE",
        "total": 35004
}
```


Successful response status code: 200

Failed response error codes: 404, 500

Get Search Results

To retrieve the results of a search job based on its job ID.

Request

`/search/jobs/{job_id}/results/`

HTTP method: GET

Attributes: See table below.

Attributes

Attribute	Type	Description
job_id	string	Required. Job ID of the search job.
offset	integer	Optional. Offset of the search results. For example, if `offset`=10 and `fetch_size`=30, then this API will return results indexed 10 to 30. **Default**: 0.
fetch_size	integer	Optional. Number of rows returned. Maximum rows: 1000. **Default**: 25.

Response

Format: JSON

Attributes: See table below.

Attributes

Attribute	Type	Description
count	integer	Number of records returned.
fields	list	Search fields that you provided in the search query in the `POST /search/jobs/` API call. For example, if you provide `event_time`, `event_meta_id`, `timestamp` as part of your search query in the `POST /search/jobs/` request, `fields` in the response would contain the event time, event ID, and the timestamp, as follows: `fields = ["event_time", "event_meta_id", "timestamp"]`

Attribute	Type	Description
has_next	boolean	Whether there are more results pages than the current page. TRUE—More results present beyond the current page. FALSE—The current page is the last page with all the results.

is_ aggregated	boolean	Whether the search results are aggregated.
		TRUE—Query is an aggregation operation. For example:
		`eventlog | where sourcetype contains \"crowdstrike\" or sourcetype contains \"edr\" | aggr count as cnt by src | sort cnt desc | limit 100"`
		FALSE—Query is not an aggregation operation. For example:
		`eventlog | fields src_ip, dest_ip`

records	list	List of records containing the fields included in the `fields` response attribute.
result_row_ count	integer	Total number of records retrieved by the search. Use `offset` and `fetch_size` to decide how many retrieved rows you access in one go.
search_end_ time	integer	End timestamp of the search, in UNIX timestamp milliseconds.
search_ start_time	integer	Start timestamp of the search, in UNIX timestamp milliseconds.

Attribute	Type	Description
status	string	Status of the search. Possible options: • `QUEUED`: Search job is waiting for execution. • `CANCELLED`: Query was canceled while it was running. • `RUNNING`: Search in progress. • `DONE`: Search query is complete and the search results are retrieved. • `FAILED`: When no result is returned due to many reasons, such as "Timeout", "Server error", "Job ID not found", or "Unexpected error". **Note:** If more than 120 seconds lapse in the `QUEUED` status, you will encounter a `FAILED` status. You can reissue a new search.
types	array	Data type of the attributes in each retrieved search record.

Examples

Request:

curl example:

```
curl -X 'GET'
    'https://api.threatstream.com/api/v1/xdr/search/jobs/6f5f5c716a8b48e1b2659b5918c
    1b51f/results/?offset=0&fetch_size=25' \
    -H 'Authorization: apikey <username>:<api_key>' \
    -H 'accept: application/json'
```

Python example:

```
import requests
response = requests.get(
```

```
'https://api.threatstream.com/api/v1/xdr/search/jobs/6f5f5c716a8b48e1b2659b5918c1b51
f/results/?offset=0&fetch_size=25',
       headers={
               'Accept': ‘application/json',
               'Authorization': 'apikey <username>:<api_key>'
       },
       params={
               'offset': 0,
               'fetch_size': 25
       }
)
```

Response:


```
{
        "count": 10,
        "fields": [
                "event_time", "sourcetype", "dcid",
                "src", "action", "dest", "url", "user",
                "event_meta_id", "_raw_id", "day", "hour", "minute"
        ],
        "has_next": false,
        "is_aggregated": false,
        "records": [
                [1727647847687, "myexamplesourcetype", "78",
                null, "allow", null, "https://www.abc-example.com/feed", null,
                8587425684, null, "29", "22", "10-0"],
                [1727647468096, "aws_cloudtrail", "1",
                "192.1.0.0", "created", "", "", "MyAdminTeam",
                8690012349, null, "29", "22", "00-0"],
        ],
        "result_row_count": 10,
        "search_end_time": 1727990933868,
        "search_start_time": 1727990924712,
        "status": "DONE",
        "types": ["timestamp", "string", "string",
                "string", "string", "string", "string", "string",
                "number", "number", "string", "string", "string"]
```

```
}
```

Successful response status code: 200

Failed response error codes: 404, 500

Chapter H: Using Lookup Tables (Classic UI)

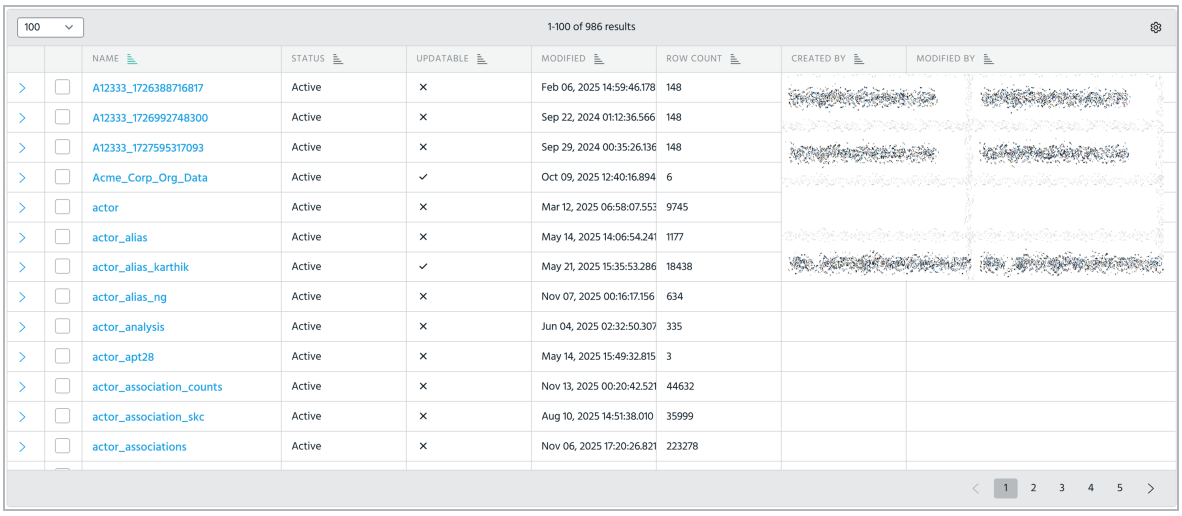
A *lookup table* is a CSV file that has enrichment data and can be any set of fields that you want to correlate with an event log field, enabling the lookup table data to enrich Search results.

For example, the eventlog lookup table empowers AQL Views and Security Analytics to map Anomali's event log schema to different event fields from varied event log sources.

Anomali offers and supports two types of lookup tables:

- **System lookup tables:** These are lookup tables that Anomali offers out of the box. See "Anomali System Lookup Tables (Classic UI)" on page 1336 for more information.
- **Custom lookup tables:** Anomali allows you to create your own lookup tables for downstream applications.
  To learn how to create a custom lookup table, refer to "Creating Custom Lookup Tables (Classic UI)" on page 1339.
  To learn how to view, append, update, export, or delete custom lookup tables, refer to "Managing Custom Lookup Tables (Classic UI)" on page 1342.

To access all lookup tables available to your organization, navigate to **Search** > **Lookup Tables**.



100 | 1-100 of 986 results

NAME	STATUS	UPDATABLE	MODIFIED	ROW COUNT	CREATED BY	MODIFIED BY
A12333_1726388716817	Active	×	Feb 06, 2025 14:59:46.178	148		
A12333_1726992748300	Active	×	Sep 22, 2024 01:12:36.566	148		
A12333_1727595317093	Active	×	Sep 29, 2024 00:35:26.136	148		
Acme_Corp_Org_Data	Active	✓	Oct 09, 2025 12:40:16.894	6		
actor	Active	×	Mar 12, 2025 06:58:07.553	9745		
actor_alias	Active	×	May 14, 2025 14:06:54.241	1177		
actor_alias_karthik	Active	✓	May 21, 2025 15:35:53.286	18438		
actor_alias_ng	Active	×	Nov 07, 2025 00:16:17.156	634		
actor_analysis	Active	×	Jun 04, 2025 02:32:50.307	335		
actor_apt28	Active	×	May 14, 2025 15:49:32.815	3		
actor_association_counts	Active	×	Nov 13, 2025 00:20:42.521	44632		
actor_association_skc	Active	×	Aug 10, 2025 14:51:38.010	35999		
actor_associations	Active	×	Nov 06, 2025 17:20:26.821	223278		

1 2 3 4 5

Search: Enter the lookup table name that you want to find.

Name: Name of the lookup table.

Status: View the activation status of the lookup table:

- Active: An active status indicates that the table has been created and is ready to be used in Search. You cannot activate a lookup table
- Inactive: An inactive status indicates that activation is not yet completed.

Updatable: Whether the lookup table is updatable.

Data Modified: Date when the lookup table was last modified.

Row Count: Number of rows in the table.

Table Settings: Select which columns to dispay.

Refresh: Refresh the list of lookup tables.

New: Create a new lookup table. Refer to "Creating Custom Lookup Tables (Classic UI)" on page 1339 for details.

User Permissions and Actions

- All users can upload lookup tables.
- All users can use Search to query any lookup table.
- Org admin users can append, update, and delete any lookup table.
- Non-org admin users can append, update, and delete only those lookup tables that they own.

Anomali System Lookup Tables (Classic UI)

The table below summarizes the lookup tables that Anomali provides out of the box.

Source	Lookup Table	Description
Your event logs	"eventlog " on page 746	Represents the schema that Security Analytics uses to map to different event log fields from varied event log sources. As an example, here are some categories of essential fields captured by the Eventlog Schema, among several others: **Network:** `event_time, action, src_ip, dest_ip, src_port, dest_port, protocol, bytes_in, bytes_out` **Web Proxy:** `event_time, action, src_ip, dest, url, user, user_agent, http_referrer, status, http_method` **DNS:** `event_time, action, src_ip, query, answer, record_type` **Email Gateway:** `event_time, message_id, subject, sender, receiver, src_ip, dest_ip` **EDR:** `event_time, event_id, event_name, evt_type, host, user, platform, process_id, file_path, file_name, image, command_line, file_hash, file_hash2, file_hash3, src`

Source	Lookup Table	Description
Security Analytics	"iocmatch" on page 1281	Generated by the Anomali Security Analytics IOC correlation engine. This engine correlates event telemetry — firewalls, proxy servers, email gateways, or endpoints, and more — with ThreatStream IOCs. The following fields are used for correlating events: **IP addresses:** `src_ip`, `dest_ip`, `src`, `dest` **Domain:** `src`, `dest`, `query`, `url` **URL:** `url`, `http_referrer` **Email:** `sender`, `receiver` **File Hash:** `file_hash, file_hash2, file_hash3`
Security Analytics	"dgamatch" on page 1255	Generated by the Anomali Security Analytics domain generated algorithm detection feature. This feature correlates domain names found in your event telemetry data with Anomali DGA intelligence.
Security Analytics	"ignite" on page 1267	Generated by the Anomali Ignite detection feature that correlates file hashes present in event logs with endpoint file hashes that identify high-fidelity intelligence.
Security Analytics	"tmmatch" on page 1305	Generated by threat model correlation that summarizes threat model matches present in your event logs.

Source	Lookup Table	Description
ThreatStream	"observables" on page 1298	Contains observable details generated from several sources such as intelligence feeds, telemetry data (SIEM, EDR data, and so on), threat bulletins, STIX/TAXII servers, and more. This lookup table features a centralized repo of threat observables from ThreatStream that supports correlation, enrichment, analysis, and integrations.
ThreatStream	"threat_models" on page 1306	Contains details about Threat Models in ThreatStream. This lookup table enables correlation and threat hunting, curates threat profiles, aiding in response prioritization, reporting, and defense planning.
Security Analytics	audit	Generates different user actions such as creating alerts, launching a forensic search, or creating a dashboard, among other actions. This lookup table helps you gain crucial insight into the history of different user actions, ensuring transparency, traceability, compliance, reporting, and accountability for every action initiated by API clients, users, or other automated processes.

Source	Lookup Table	Description
ThreatStream	"consolidated_actor_descriptions_ng" on page 1241	Adds AI-generated descriptions to the AI Actor Profile dashboard for each unique threat actor and its aliases in ThreatStream.
ThreatStream	"consolidated_actor_observables_ng" on page 1245	Links threat actor associations from the `consolidated_actor_profiles_ng` lookup table with corresponding observable information based on Actor IDs. This table contributes data to the AI Actor Profile dashboard.
ThreatStream	"consolidated_actor_profiles_ng" on page 1250	Populates the AI Actor Profile dashboard with consolidated information about threat actors, including aliases, associations, motivations, and other contextual details available in ThreatStream.
ThreatStream	"actor_associations" on page 722	Generates information about threat actor associations available in your ThreatStream instance.

Creating Custom Lookup Tables (Classic UI)

You can add a lookup table to your AQL Search environment by uploading a comma-separated value (CSV) file. Additionally, you can add values to existing lookup tables.

CSV Import Guidelines

To successfully upload a lookup table, follow these file upload guidelines:

- The upload file must have a heading row with the lookup table field names.
- Avoid using the same field names as the eventlog schema.

- In searches, field name is case sensitive, so use capitalization conventions that are easy for search users to follow. For example, all lowercase.
- Enclose values that have spaces in straight quotation marks (").
- File formats must be CSV, .gz, or .tar.
- The upload file can be a maximum of 100 MB.

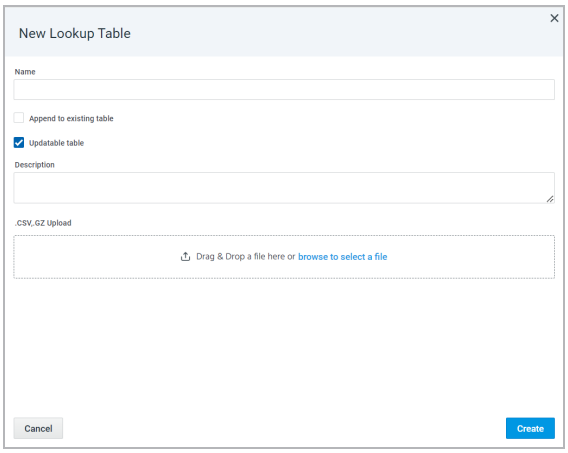
> **Note:**
> If your CSV file is larger than 100 MB, use the following guidelines to upload your data:
>
> - Compress the file to reduce its size.
> - Split the content into multiple smaller files. After that, upload the first file to create a lookup table and append the remaining files to this table. See "Appending to an Existing Lookup Table" on page 1343 for more information.

The following is an example of a lookup table:

	A	B	C	D	E
1	geo_ip	hostname	country	city	zipcode
2	192.0.2.1	tn-1-host	CA	Vancouver	"V6H 3K1"
3	198.51.100.1	tn-2-host	US	San Francisco	94117
4	203.0.113.0	tn-3-host	MX	Guanajuato	36050
5					

To upload a lookup table:

1. Create a CSV file that contains the required values, as described above.
2. Navigate to **Manage** > **Lookup Tables**.
3. Click **New**. The New Lookup Table dialog box opens.



4. Enter a table name in the **Name** field.

> **Note:**
> - This is the lookup table name that users can reference in AQL Search.
> - The name may contain only alphanumeric characters and underscores.
> - The name cannot begin with a number.
> - Do not name the table the same name as an event field or lookup table field.

5. (Optionally) Select **Append to existing table** if you want to add the rows to an existing lookup table. See "Appending to an Existing Lookup Table" on page 1343 for details.

6. (Optionally) Select **Updatable table** if you want to be able to make updates to the table. See "Updating Lookup Tables" on page 1344 for details.

> **Note:** System tables and lookup tables created using the `yieldtable` operator cannot be updated.

7. (Optionally) Enter a description of the lookup table in the Description box.

8. Use the file controls to browse and select your CSV file.

9. Click **Create**.

The lookup table is added to the list of lookup tables on the Lookup Tables list page.

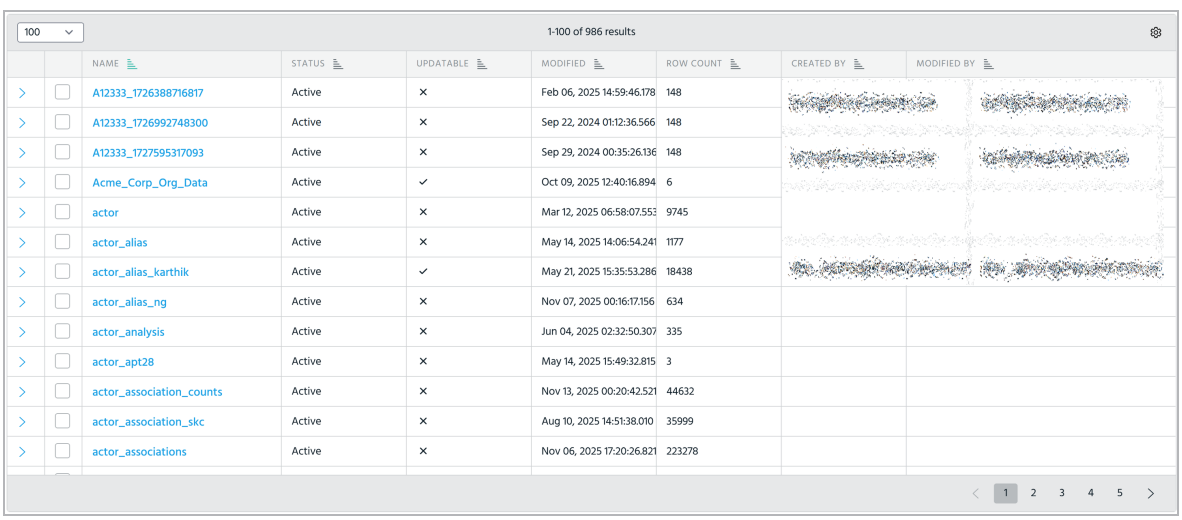
Managing Custom Lookup Tables (Classic UI)

From the **Search** > **Manage** > **Lookup Tables** page, you can perform the following actions with custom lookup tables:

- Create a lookup table
- Append to an existing lookup table
- View lookup tables
- Update a lookup table
- Export a lookup table
- Delete a lookup table

Lookup table validation

If there is a problem with the table validation, the status of the table turns to Failed. To know the reason the lookup table activation failed, click **View Details** next to the Failed status.

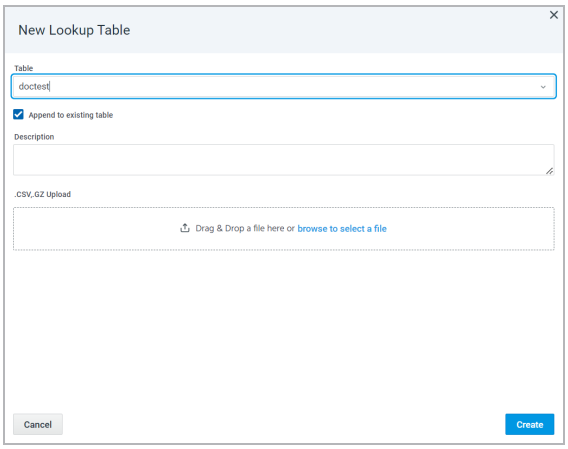


Appending to an Existing Lookup Table

> **Note:** This feature only enables you to add rows in bulk to an existing lookup table. To delete rows in an existing non-updatable lookup table, you must delete the table and upload a new table with the rows you need. To add, edit, or delete rows in an updatable lookup table, use the update feature ( ) on the Lookup Table tab. Refer to "Updating Lookup Tables" on the next page for details.

To append (add rows in bulk) to an existing lookup table:

1. Create a CSV file that contains the rows you want to add to an existing lookup table. Note that the rows you add must have the same number of columns as the lookup table to which you add the rows.
2. Navigate to **Manage** > **Lookup Tables**.
3. Click the **New**. The New Lookup Table dialog box opens.
4. Click **Append to existing table**.



5. Select the lookup table to which you want to add rows.
6. (Optionally) Enter a description of the lookup table in the Description box.
7. Use the file controls to browse and select your CSV file.
8. Click **Create**.

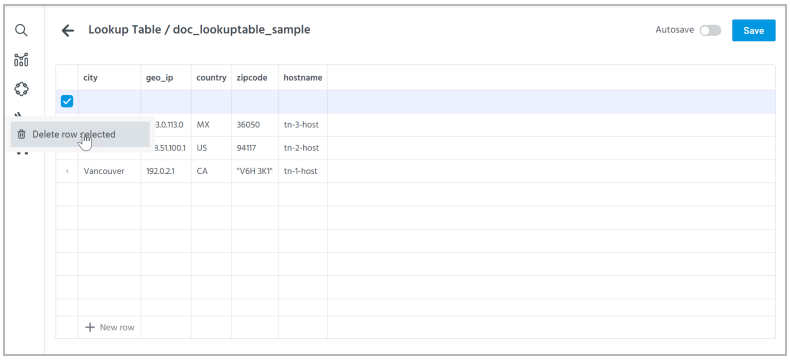
The rows are added to the existing lookup table.

Updating Lookup Tables

Updatable lookup tables can be modified. You can edit cell values, add rows, and delete rows.

To update a lookup table:

1. Navigate to **Manage** > **Lookup Tables**.
2. Hover your mouse over the lookup table you want to update and click the Edit icon ( ). You are redirected to the table in the edit mode.



3. (Optionally) Disable **Autosave** if you don't want to automatically save your updates. When **Autosave** is enabled, updates are auto-saved every minute.
4. Modify the lookup table:
   - To add a new row to the table, click **+ New row**.
   - To delete a row, select the row you want to delete, right click on the selected row, and click **Delete row selected**.
     To delete all rows, select the header row, right click on the rows, and click **Delete rows 1-x**.
   - To edit a cell value, double click on a cell and edit its value. Values in the header row cannot be modified.
5. If **Autosave** is disabled, click **Save**.

The lookup table is updated.

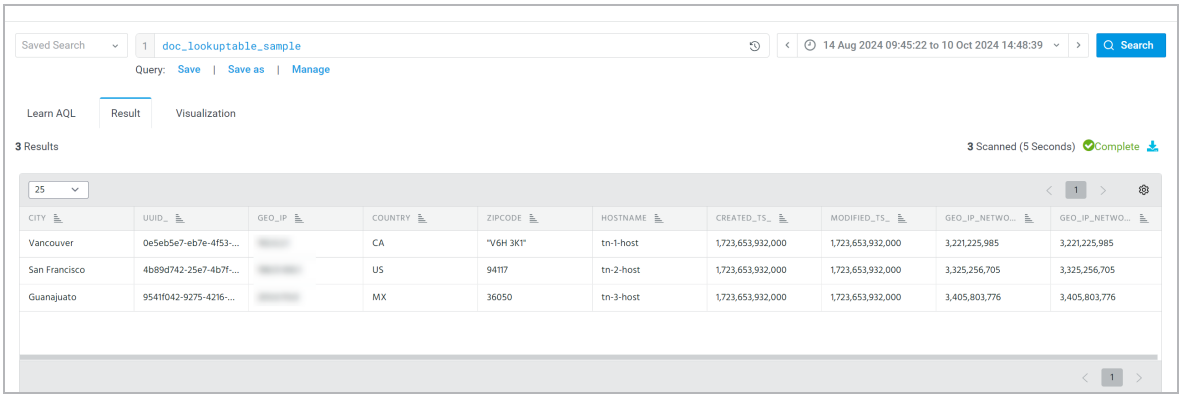
Viewing Lookup Tables

You can view all lookup tables in Search on the Result tab.

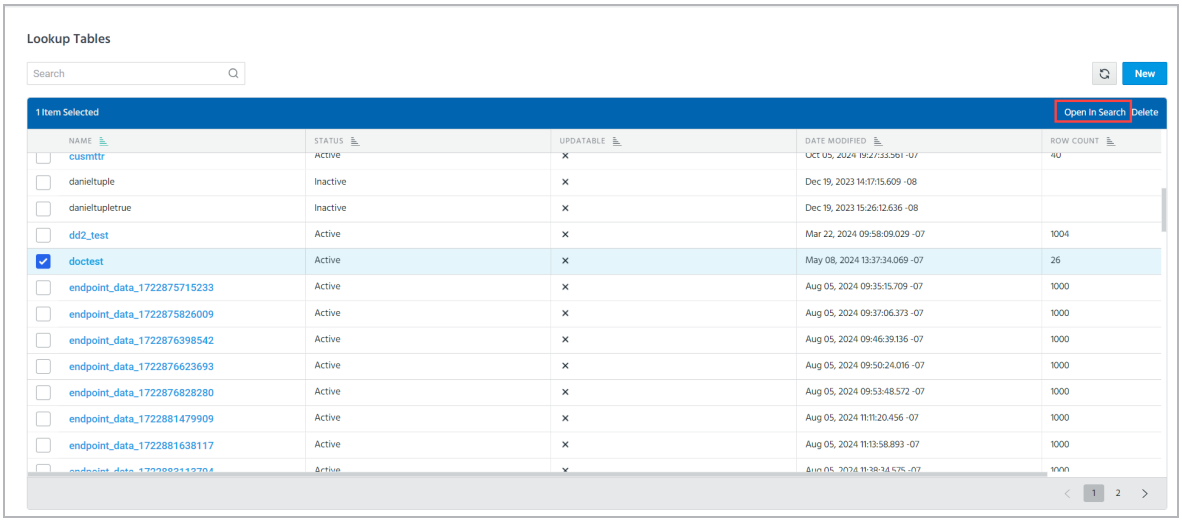
To view a lookup table:

1. Navigate to **Manage** > **Lookup Tables**.
2. Click the active lookup table of interest.

You are redirected to the Result tab of the AQL Search page.



Alternately, to view a lookup table in Search, select a lookup table of your interest and click **Open in Search**. You are redirected to the Result tab of the AQL Search page.

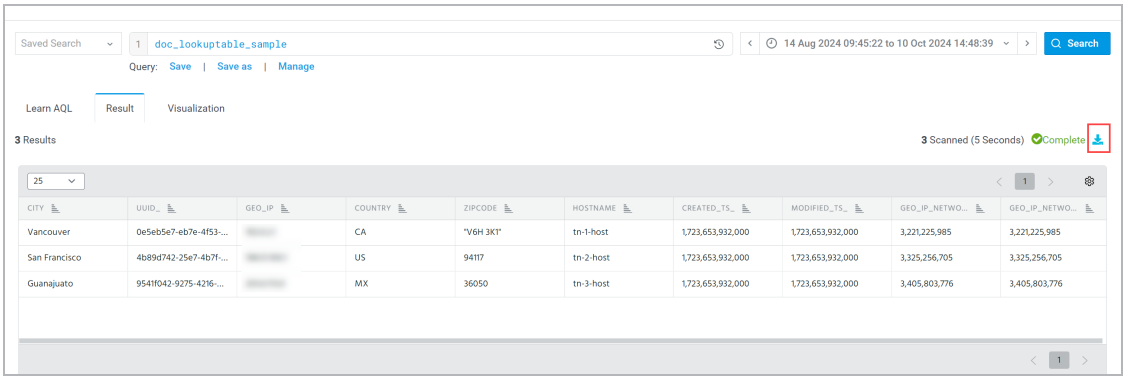


Exporting Lookup Tables

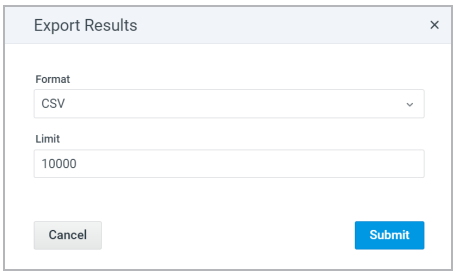
Lookup tables can be exported in CSV or JSON format.

To export a lookup table:

1. Navigate to **Manage** > **Lookup Table**.
2. Click the active lookup table of interest. You are redirected to the Result tab of the AQL Search page.
3. Click the Export icon ( ).



4. In the Export Results dialog box, select an export format (CSV or JSON) and enter a maximum number of rows you want to export.



5. Click **Submit**.

Downloading of the lookup table starts immediately.

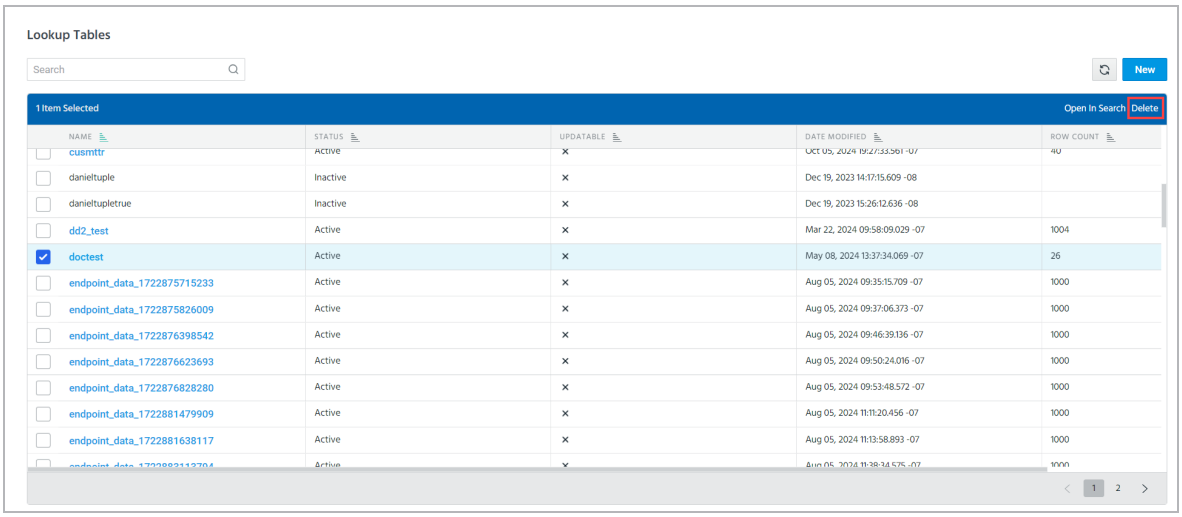
Deleting Lookup Tables

To delete a lookup table:

1. Navigate to **Manage** > **Lookup Tables**.
2. Mouse over the lookup table you want to delete and click the Delete icon ( ).

The lookup table is deleted.

Alternately, to delete a lookup table, select a lookup table of your interest and click **Delete**.



Chapter I: Using Macros (Classic UI)

This section includes the following topics:

Creating a Macro (Classic UI) 1349
Managing Custom Macros (Classic UI) 1351
Using Custom Macros in Search (Classic UI) 1353
Viewing Custom Macros (Classic UI) 1355
Using ThreatStream Macros in Search (Classic UI) 1356
Available Macros 1357

A macro is a reusable, modular piece of an AQL search query that you can insert into other searches. It is designed to help you avoid manually constructing lengthy strings of parameters for complex search queries. You can use pre-built, out-of-the-box ThreatStream macros or create custom macros tailored to your organization's needs. Custom macros created within your organization can be reused in Search, edited, cloned, and deleted.

Key Benefits of Macros

Macros provide the following search benefits:

- **Simplicity**

  You can significantly reduce the complexity of search queries by using macros.

- **Maintainability**

  You can automatically update all search queries that include macros by configuring only macro definitions (custom macros only).

- **Consistency**

  You can reduce the amount of search errors due to the same logic used in macros across various queries.

For details on creating custom macros, refer to "Creating a Macro (Classic UI)" on the next page.

For details on accessing your organization macros, refer to "Viewing Custom Macros (Classic UI)" on page 1355.

For details on managing custom macros, refer to "Managing Custom Macros (Classic UI)" on page 1351.

For details on using macros in Search, refer to "Using Custom Macros in Search (Classic UI)" on page 1353.

For details on using out-of-the-box macros in Search, refer to "Using ThreatStream Macros in Search (Classic UI)" on page 1356.

Creating a Macro (Classic UI)

On the Macros page, you can create new custom macros that will be available to all users of your organization. Macros can include any part of a search query, such as a `field` list or `where` clause, and do not need to be a complete query. Each macro within the Anomali platform requires a unique name and the AQL definition. The description and arguments are optional.

To create a macro:

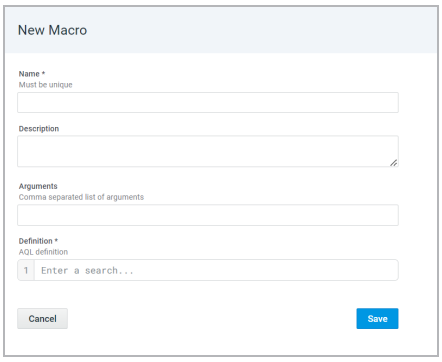
1. Navigate to **Manage** > **Macros**.
2. Click **New**.

3. In the New Macro dialog box that opens, provide the following information:
   - **Name:** Enter a unique name for the macro. To be used in Search, only the following characters are accepted in macro names: letters, numbers, underscores (_), hyphens (-), and dots (.). However, the first character of a macro name must be a letter.

> **Note:** If a macro includes arguments, add the number of arguments in parenthesis after the macro name. For

> example, `test_macro(3)`, where `(3)` is the number of arguments in the macro.

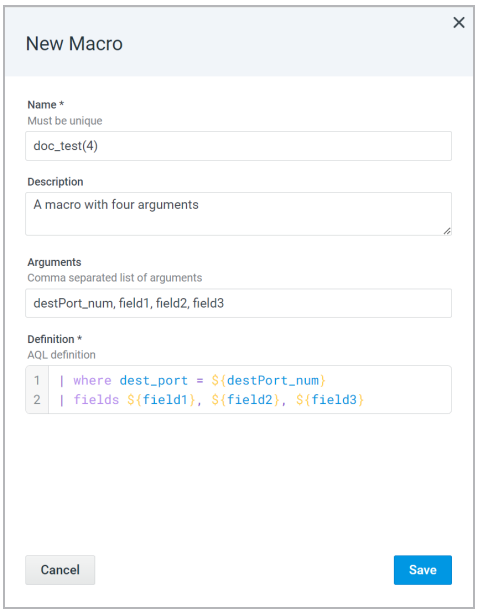
- (Optional) **Description:** Enter a description of what the macro can accomplish in Search.
- (Optional) **Arguments:** Add macro arguments when you need to change or control the output of the macro. Arguments are the actual values that are passed to the macro when it is executed. For example, to search for event traffic going to destination port 443, the search query will be the following: `|where dest_port = 443`. However, using an argument instead of `443` allows any port number to be inserted into the query.
- **Definition:** Enter the AQL search query that will be executed when the macro is inserted in Search. Arguments must be wrapped in `${ }` in the macro definition.



4. Click **Save**.

The newly created macro is added to the list of custom macros.

Below is an example of creating the `doc_test(4)` macro with four arguments where `destPort_num` is an argument for the destination port, and `field1, field2, and field3` are arguments that act as a placeholders for the fields.

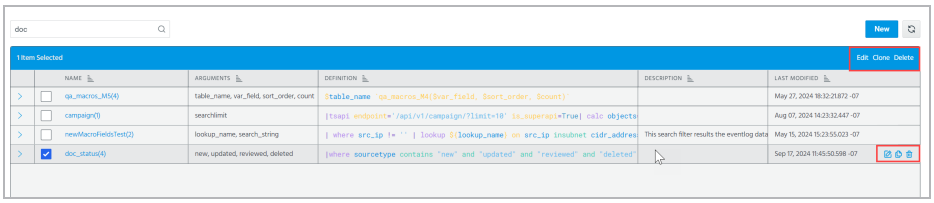


To learn how to use macros with arguments in Search, refer to "Using Custom Macros in Search (Classic UI)" on page 1353.

Managing Custom Macros (Classic UI)

Custom macros created within your organization can be edited, cloned, and deleted on the Macros page. These actions can be taken in two ways:

- By selecting a macro and clicking **Edit**/**Clone**/**Delete**.
- By hovering your mouse over a macro and clicking the edit/clone/delete icon.



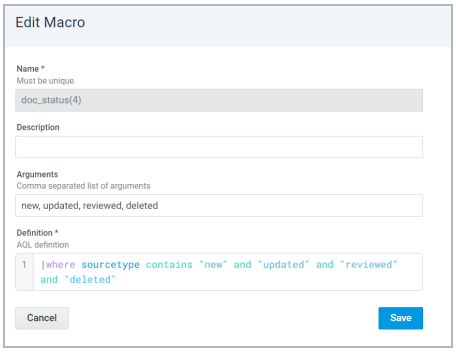
Editing a Macro

When editing a macro, you can change its the description, arguments, and definition. The name cannot be modified. To change the name, clone the macro. See "Cloning a Macro" on the next page for details.

> **Note:** Once a macro is changed, the changes are reflected in alerts, saved searches, and dashboards which were created with AQL search queries referencing this macro.

To edit a macro:

1. Navigate to **Manage** > **Macros**.
2. Select the macro you want to edit and click **Edit**. Alternatively, hover the mouse over the macro you want to edit and click .
3. In the Edit Macro dialog box that opens, make the required changes.



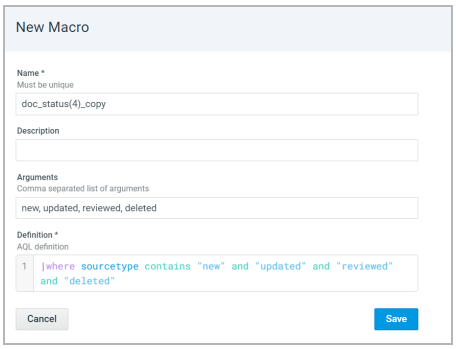
4. Click **Save**.

The changes are applied to the macros.

Cloning a Macro

To clone a macro:

1. Navigate to **Manage** > **Macros**.
2. Select the macro you want to clone and click **Clone**. Alternatively, hover the mouse over the macro you want to clone and click .
3. In the New Macro dialog box that opens, configure the macro name, description, arguments, and definition if necessary.



4. Click **Save**.

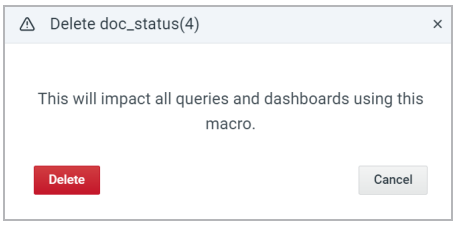
The cloned macro is added to the list of macros.

Deleting a Macro

To delete a macro:

> **Note:** This action impacts all AQL queries and dashboards that use the macro that you are going to delete.

1. Navigate to **Manage** > **Macros**.
2. Select the macro you want to delete and click **Delete**. Alternatively, hover the mouse over the macro you want to delete and click .
3. In the warning message that opens, click **Delete**.



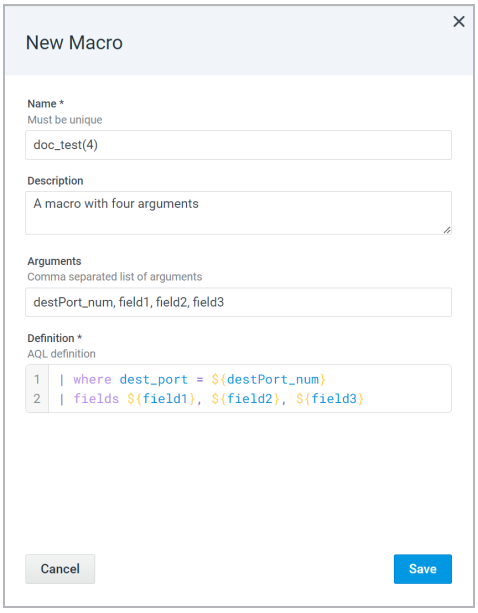
The macro is deleted.

Using Custom Macros in Search (Classic UI)

Macros are entered in the Search field with a back tick character (`) before and after the macro name. For example, `my_macro(2)`. Any macro can be referenced

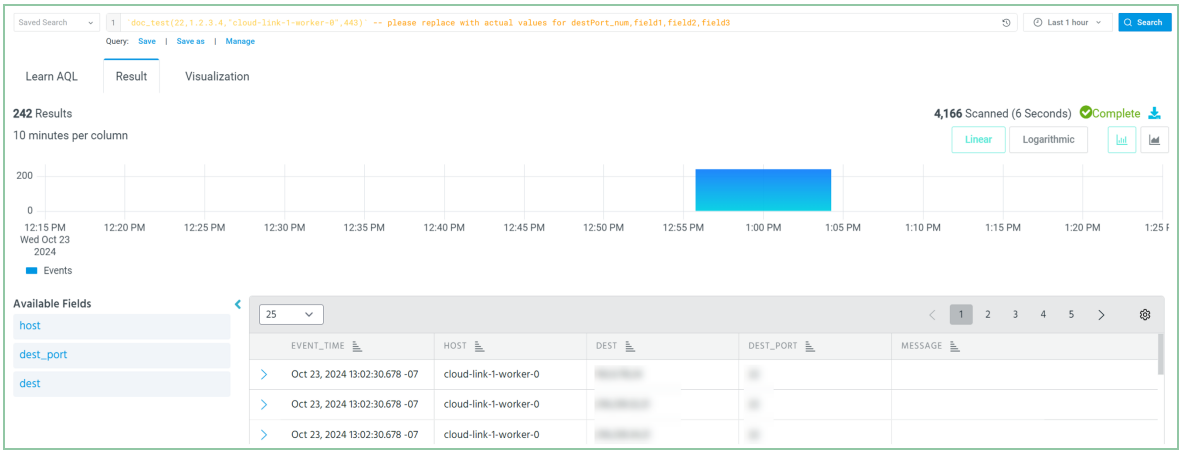
within other macros using the same syntax.

To invoke a macro with arguments, replace arguments with actual values in the order in which they are defined within the macro. For example, in the `doc_test(4)` macro, `destPort_num`, `field1`, `field2`, and `field3` are defined as arguments.



To run the `doc_test(4)` macro in Search, replace the `destPort_num`, `field1`, `field2`, and `field3` arguments with the actual values. The values must be entered in the same order as the text in the arguments. Therefore, the first value is the destination port followed by the fields. For example, `` `doc_test(22,1.2.3.4,"cloud-link-1-worker-0","optic")` ``.

Below is an example of the `doc_test(4)` macro used in Search.



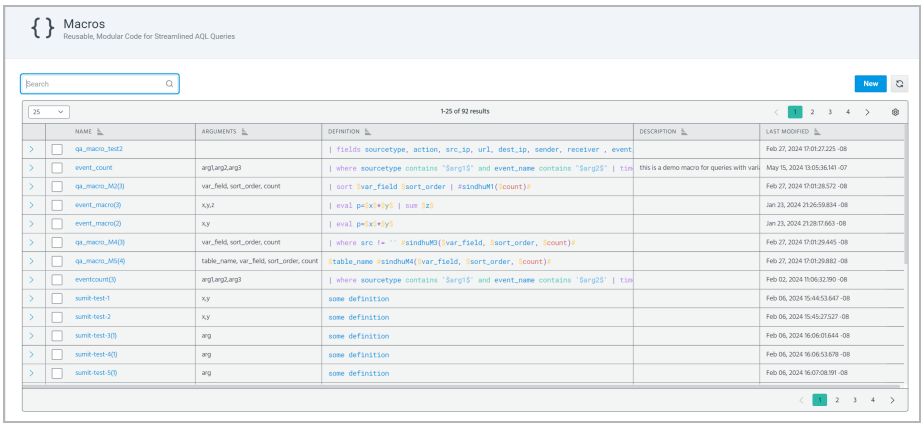
If your argument value is of the string datatype, enclose it in double quotes ("").

In the example above, the argument `evt_msg` takes the string value `Bundle stats` enclosed in double quotes.

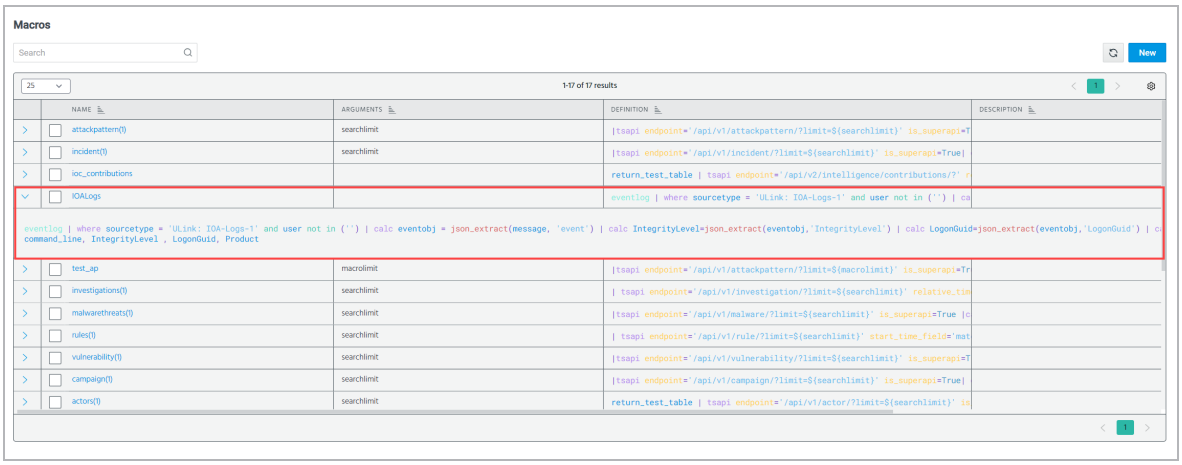
Viewing Custom Macros (Classic UI)

On the Macros page, you can view all macros created by users of your organization. Macros created within your organization can be referenced in multiple searches without the need to rewrite the entire piece of query.

To view the list of custom macros, navigate to **Manage** > **Macros**.



To view a macro definition, click the arrow next to the macro of your interest.

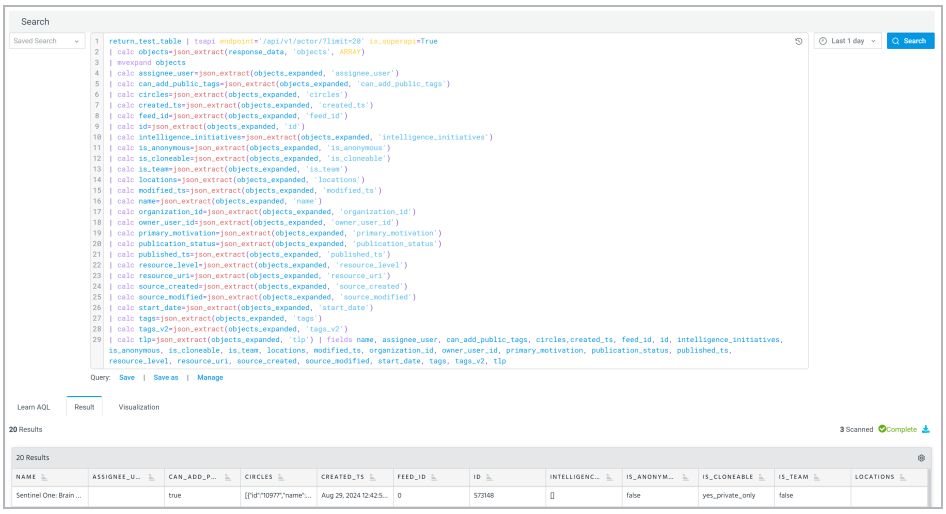


On the Macros page, you can take the following actions:

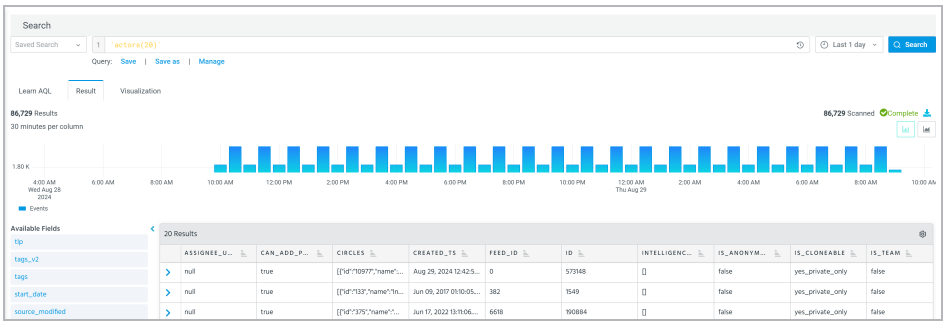
- Create new macros. See "Creating a Macro (Classic UI)" on page 1349 for details.
- Edit custom macros. See "Editing a Macro" on page 1351 for details.
- Clone custom macros. See "Cloning a Macro" on page 1352 for details.
- Delete custom macros. See "Deleting a Macro" on page 1353 for details.

Using ThreatStream Macros in Search (Classic UI)

Anomali offers out-of-the-box macros that enable you to fetch and analyze ThreatStream data without building complex search queries. For example, the image below demonstrates the search query that returns 20 most recent Actors and their field details.

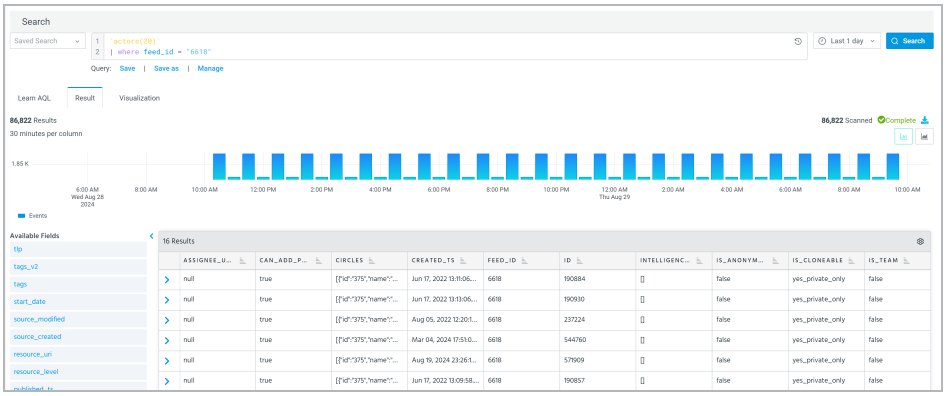


To avoid building such a complex search query, you can use the `actors(20)` macro to return the same results.



Similar to custom macros, out-of-the-box macros are entered in the Search field with a back tick character (`) before and after the macro name. For example, `ioc_feeds`, `malwarethreats(50)`, `rules(30)`.

After running a search query with a macro, you can continue analyzing and filtering data displayed on the Result tab. This can be done by building conditions with available macro fields. See "Available Macros" below for more information about available macros and their fields.



Out-of-the-box macros can be used independently or within a search query.

Search queries containing macros can be saved and managed the same way as regular search queries. See "Anomali Search" on page 21 for details.

Additionally, they can be used to create custom dashboards. See Creating a Dashboard for details.

Available Macros

Use this section as a reference for all out-of-the box macros available in Search. Note that some macros require you to specify a limit for the number of returned

results. The maximum limit you can specify is 1,000. The results are returned based on modified timestamp.

Available macros can be cloned or copied, but they cannot be modified.

actors (limit)

This macro allows you to query the data related to Actors. When used in Search, it returns the fields containing Actor details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	User assigned to this Actor
can-add_ public_tags	Indicates whether the assigned user can add public tags or not. Possible values: true, false.
circles	Trusted circles with which the Actor is shared.
created_ts	UTC time stamp of when the Actor was first created in ThreatStream. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35
feed_id	Numeric ID of the threat feed that provided the threat model entity.
id	ID of the Actor. This ID is automatically assigned to the entity by ThreatStream at the time of creation. You can use this ID to associate the Actor with other threat model entities.
intelligence_ initiatives	Intelligence Initiatives associated with the Actor.
is_anonymous	Indicates whether the Actor user and organization information is anonymized. is_anonymous=1 for true (anonymous) is_anonymous=0 for false (not anonymous)
is_cloneable	Indicates whether the Actor entity can be cloned.

Fields	Description
is_team	Indicates whether the Actor is a team or known to work as an individual (as specified in the Team check box in the UI). True—Actor is a team False—Actor is an individual Default: False
locations	locations applied to the Actor. Possible values include letter codes of regions, countries, and administrative areas (US States and Washington DC only) as defined by STIX 2.1. For example, eastern-africa, central-asia, europe, africa, NL, GB, US-FL, and US-MO.
modified_ts	When the Actor was last updated on the ThreatStream Cloud platform. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35
name	Name of the Actor. Actor name is associated with your organization. Therefore, the name must be unique within your organization. However, two Actors with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that owns the Actor.
owner_user_id	ID of the ThreatStream user that created the Actor.
primary_ motivation	Actor's primary motivation
publication_ status	An Actor can be in the new, pending_review, review_ requested, reviewed, published statuses.

Fields	Description
published_ts	Time stamp of when the entity was published on ThreatStream, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_level	For internal use only.
resource_uri	The API URL to the Actor information.
source_created	Timestamp to when the Actor was first created.
source_ modified	Timestamp to when the Actor was last updated.
start_date	Specify the time when the Actor was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Actor. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Actor.
tlp	Traffic Light Protocol assigned to the Actor. Possible values: red, amber, amber+strict, green, and clear.

attackpattern(limit)

This macro allows you to query the data related to Attack Patterns. When used in Search, it returns the fields containing Attack Pattern details, which then can be included in the operators for further analysis.

Fields	Description
aliases	Other names by which the Attack Pattern is known.
assignee_user	ID of the user to which the Attack Pattern is assigned
can_add_public_tags	Indicates whether a user can add public tags to the Attack Pattern.
circles	All trusted circles to which the Attack Pattern belongs.
created_ts	Time stamp of when the Attack Pattern was created on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Attack Pattern.
id	ID of the Attack Pattern. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Attack Pattern.
is_anonymous	Indicates whether the Attack Pattern user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Attack Pattern can be cloned.
is_mitre	Indicates whether the Attack Pattern is described in the MITRE list. True—if the Attack Pattern is described False—if the Attack Pattern is not described

Fields	Description
is_public	Indicates whether the Attack Pattern is public or private (including belonging to a trusted circle). False—if the Attack Pattern is private or belongs to a trusted circle True—if the Attack Pattern is public
is_team	Indicates whether the Attack Pattern is a team or known to work as an individual. True—Attack Pattern is a team False—Attack Pattern is an individual
killchainstages	List of attack structure phases. Values: `"action-on-objectives", "command-and-control", "delivery", "exploitation", "installation", "reconnaissance", "weaponization"`
locations	Locations applied to the Attack Pattern. Possible values include letter codes of regions, countries, and administrative areas (US States and Washington DC only) as defined by STIX 2.1. For example, eastern-africa, central-asia, europe, africa, NL, GB, US-FL, and US-MO.
modified_ts	Timestamp of when the Attack Pattern was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Attack Pattern. The Attack Pattern name is associated with your organization. Therefore, the name must be unique within your organization. However, two Attack Patterns with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.

Fields	Description
org_id	ID of the organization that created the entity on ThreatStream.
owner_user_id	ID of the user that created the Attack Pattern on ThreatStream.
primary_motivation	Primary motivation of the Attack Pattern.
publication_status	Attack Pattern entities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Timestamp of when the Attack Pattern was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_level	
resource_uri	URI of the Aattack Pattern entity on ThreatStream
source_created	Timestamp to when the Attack Pattern was first created.
source_modified	Timestamp to when the Attack Pattern was last updated.
system	System associated with the Attack Pattern.
start_date	Specify the time when the Attack Pattern was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Attack Pattern. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.

Fields	Description
tags_v2	Additional information on the tags assigned to the Attack Pattern.
target_industry	Target industries associated with the Attack Pattern.
tlp	Traffic Light Protocol assigned to the Attack Pattern. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Attack Pattern for STIX compliance.
workgroups	Workgroups to which the Attack Pattern is visible.

campaign(limit)

This macro allows you to query the data related to Campaigns. When used in Search, it returns the fields containing Campaign details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the Campaign is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Campaign.
circles	All trusted circles to which the Campaign belongs.
created_ts	Time stamp of when the Campaign was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
end_date	The time when this Campaign is known to have ended. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.

Fields	Description
feed_id	Numeric ID of the threat feed that provided the Campaign.
id	ID of the Campaign. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Campaign.
is_anonymous	Indicates whether the Campaign user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Campaign can be cloned.
is_public	Whether the Attack Pattern is public or private (including belonging to a trusted circle). False—if the Attack Pattern is private or belongs to a trusted circle True—if the Attack Pattern is public
modified_ts	Time stamp of when the Campaign was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Campaign. The Campaign name is associated with your organization. Therefore, the name must be unique within your organization. However, two campaigns with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Campaign on ThreatStream.

Fields	Description
owner_user_id	ID of the user that created the Campaign on ThreatStream.
publication_status	Campaign entities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Time stamp of when the Campaign was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Campaign entity on ThreatStream.
source_created	Timestamp to when the Campaign was first created.
source_modified	Timestamp to when the Campaign was last updated.
start_date	Specify the time when the Campaign was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Campaign. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Campaign.
target_industry	Target industries associated with the Campaign.
tlp	Traffic Light Protocol assigned to the Campaign. Possible values: red, amber, amber+strict, green, and clear.

Fields	Description
uuid	UUID (universally unique identifier) assigned to the Campaign for STIX compliance.
workgroups	Workgroups to which the Campaign is visible.

incident(limit)

This macro allows you to query the data related to Incidents. When used in Search, it returns the fields containing Incidents details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the Incident is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Incident.
circles	All trusted circles to which the Incident belongs.
created_ts	Time stamp of when the Incident was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
end_date	The time when this Incident is known to have ended. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Incident.
id	ID of the Incident. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Incident.

Fields	Description
is_anonymous	Indicates whether the Incident user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Incident can be cloned.
is_public	Indicates whether the Incident is public or private (including belonging to a trusted circle). False—if the Incident is private or belongs to a trusted circle True—if the Incident is public
modified_ts	Timestamp of when the Incident was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Incident. The Incident name is associated with your organization. Therefore, the name must be unique within your organization. However, two Incidents with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Incident on ThreatStream.
owner_user_id	ID of the user that created the Incident on ThreatStream.
publication_status	Incident entities can be in new, pending_review, review_requested, reviewed, or published statuses.

Fields	Description
published_ts	Timestamp of when the Incident was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Incident entity on ThreatStream.
source_created	Timestamp to when the Incident was first created.
source_modified	Timestamp to when the Incident was last updated.
start_date	Specify the time when the Incident was first known to be active. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the Incident. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Incident.
target_industry	Target industries associated with the Incident.
tlp	Traffic Light Protocol assigned to the Incident. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Incident for STIX compliance.
workgroups	Workgroups to which the Incident is visible.

investigations

This macro allows you to query statistics of the investigations accessible via Anomali platform. When used in Search, it returns the fields containing information about investigations, which then can be included in the operators for further analysis.

Fields	Description
created_ts	Timestamp when the investigation was created.
description	Short description of the investigation.
id	Unique investigation ID. This identifier is assigned to the investigation when it is created on ThreatStream.
modified_ts	Timestamp when the investigation was last modified.
name	Investigation name.
priority	Priority assigned to the investigation. Possible values: very low, low, medium, high, very high
status	Status assigned to the investigation. Possible values: completed, in-progress, pending, unassigned
tags	Tags associated with the investigation.
tlp	TLP setting for the investigation. In the context of investigations, the TLP color sets the maximum level of information that users outside your organization have access to when viewing your investigation.
assignee_id	ID of the user or workgroup to which the investigation is assigned.
assignee_type	Type of assignee: "user" or "tsworkgroup"
elements	Number of entities associated with the investigation.

Fields	Description
r_type	Type of entity associated with the investigation. Possible values listed below. l Actors:actor l Campaigns:campaign l Incidents:incident l Observables:intelligence2 l Signatures:signature l ThreatBulletins:tipreport l TTP:ttp l Vulnerabilities:vulnerability
r_id	Unique ID assigned to the entity.
add_related_ indicators	When enabled, observables related to the entity you are associating with the investigation are also added. add_related_indicators=1 for true (add) add_related_indicators=0 for false (do not add)
tips	For internal use only.
workgroups	Workgroups with which the Investigation is shared.
assignee	User or workgroup to which the investigation is assigned.
circles	Trusted circles with which the investigation is shared.
intelligence_ initiatives	Intelligence Initiatives associated with the Investigation
owner_org	Organization of the owner. In most cases, this is the same as the assignee organization.
owner_org_id	Organization ID of the owner. In most cases, this ID is the same as the assignee organization ID.
resource_uri	Internal URL for the Investigation within ThreatStream.
source_type	Source from which the investigation was created.

ioc(limit)

This macro allows you to query details of observables. When used in Search, it returns the fields containing observable details, which then can be included in the operators for further analysis.

Attribute	Description
asn	Autonomous System (AS) number associated with the indicator.
can_add_ public_tags	Indicates whether a user can add public tags to an observable.
confidence	Level of certainty that an observable is of the reported indicator type. Confidence scores range from 0-100, in increasing order of confidence, and is assigned by ThreatStream.
country	Country associated with theobservable.
created_by	User that created the intelligence.
created_ts	When the observable was first seen on the ThreatStream cloud platform. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35.
description	This attribute is for internal use and can be ignored.
detail2	This attribute is for internal use and can be ignored.
expiration_ts	Time stamp of when intelligence will expire on ThreatStream, in UTC time.
feed_id	Numeric ID of the threat feed that generated the observable. feed_id = 0 for user-created observables.

Attribute	Description
id	Unique ID of the observable. This identifier is assigned to the observable when it is first created on ThreatStream. Unlike update_id, this identifier never changes as long as the observable is available on ThreatStream.
ip	The IP address associated with the observable if the imported indicator is a domain or a URL.
is_anonymous	Whether the organization and user information is anonymized when the observable is accessed by users outside of the owner organization.
is_editable	Indicates whether the imported entity can be updated by an intelligence source. This attribute is reserved for intelligence source providers and can be ignored.
is_public	Visibility of the observable—public or private. 0/False—if the indicator is private or belongs to a Trusted Circle 1/True—if the indicator is public **Default:** 0/False **Note:** This attribute is treated as a string or boolean in requests and as boolean in responses.
itype	Indicator type.
latitude	Latitude associated with the Geo location of the IP.
longitude	Longitude associated with the Geo location of the IP.
meta	Tags associated with the observable. These tags can be used to search for related observables.

Attribute	Description
modified_ts	When the indicator was last updated on the ThreatStream Cloud platform. Date must be specified in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time, in UTC time. For example, 2014-10-02T20:44:35
name	Name of the observable.
org	Registered owner (organization) of the IP address associated with the observable.
owner_ organization_id	ID of the organization that brought in the observable, either through a threat feed or through the import process.
rdns	Domain name (obtained through reverse domain name lookup) associated with the IP address that is associated with the observable.
retina_ confidence	Confidence score assigned to the observable by Anomali machine learning algorithms.
severity	Severity score of the observable.
sort	This attribute is for internal use only and can be ignored.
source	Usually an obfuscated description of the source of the observable; the description may not be obfuscated if the user belongs to your organization. Examples of source names: threat feed name, Analyst (if the user name is obfuscated).
source_ locations	Source locations associated with the indicator. Possible values include letter codes of regions, countries, and administrative areas (US States and Washington DC only) as defined by STIX 2.1. For example, `eastern-africa`, `central-asia`, `europe`, `africa`, `NL`, `GB`, `US-FL`, and `US-MO`.

Attribute	Description
source_ reported_ confidence	A risk score from 0 to 100, provided by the source of the observable.
status	Status assigned to the observable. For example, active, inactive, falsepos.
subtype	For hash observables—those with type=md5—subtype provides additional metadata on the type of hash associated with the observable. Possible values: MD5, SHA1, SHA256, SHA512.
tags	Tags applied to the specified observables.
target_industry	Target industries associated with the observable.
threat_type	Summarized threat type of the observable. For example, malware, compromised, apt, c2, and so on.
tlp	Traffic Light Protocol designation for the intelligence—red, amber, amber+green, clear.
trusted_circle_ ids	IDs of the trusted circles with which the indicator is shared.
type	Observable type—domain, email, ip, md5, string, url.
update_id	An incrementing numeric identifier associated with each update to intelligence on ThreatStream. Every time observable data changes on ThreatStream (for example, if it is updated or expires), the update_id of the indicator is incremented and stored in the observable. This identifier is mostly used by software that interact with ThreatStream to download intelligence, such as ThreatStream Link and ThreatStream APIs.
uuid	UUID (universally unique identifier) assigned to the observable for STIX compliance.

Attribute	Description
value	Value of the observable. For example, value = <192.168.0.10> or value = <http://www.google.com>.
workgroups	Workgroups associated with the observable.

ioc_contributions

This macro allows you to query statistics of the intelligence contributed to by your organization and community. When used in Search, it returns the fields containing information about contributed observables, which then can be included in the operators for further analysis.

Read more...

Fields	Description
community	Total number of observables contributed by the community.
community_ delta	The delta of the number of observables contributed by the community in the specified timespan as compared to the previous timespan. The "-" sign is used to indicate that less observables were contributed than during the previous timespan.
organization	Total number of observables contributed by the your organization.
organization_ delta	The delta of the number of observables contributed by your organization in the specified timespan as compared to the previous timespan. The "-" sign is used to indicate that less observables were contributed than during the previous timespan.
total	Total number of the contributions from your organization and community.

ioc_feeds

This macro allows you to query statistics of the observables imported from various feeds. When used in Search, it returns the fields containing information about feeds and observables, which then can be included in the operators for further analysis.

Fields	Description
total	Total number of observables contributed by the feed.
observables	Total number of observables contributed by the feed in the selected timespan.
feed_name	Name of the feed.
feed_categoty	This attribute is for internal use and can be ignored.
observables_ previous_period	Number of observables imported in the previous import. Not available for manual imports.
feed_id	The ID number for the feed
falsepositives	The Number of false positives in the observables in the selected timespan.

malwarethreats(limit)

This macro allows you to query statistics of the malware threats accessible via platform. When used in Search, it returns the fields containing information about malware, which then can be included in the operators for further analysis.

Fields	Description
created_ts	Time stamp of when the malware was created on ThreatStream, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
uuid	UUID (universally unique identifier) assigned to the malware for STIX compliance.
id	ID of the malware entity. This ID is automatically assigned to the entity by ThreatStream at the time of creation. You can use this ID to associate the malware entity with other Threat Model entities.

Fields	Description
modified_ts	Time stamp of when the malware was last updated on ThreatStream, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the malware. Malware name is associated with your organization. Therefore, the name must be unique within your organization. However, two malware with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
tags_v2	Additional information on the tags assigned to the Actor.
source_modified	Time stamp of when the malware was last modified by the source, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
tags	Tags assigned to the malware. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing,exploitation.
tlp	Traffic Light Protocol designation for the malware—red, amber, amber+strict, green, clear.

Fields	Description
source_created	Time stamp of when the malware was originally created by the source, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
published_ts	Time stamp of when the malware was originally published by the source, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
publication_ status	Malware entities can be in new, pending_review, review_ requested, reviewed, or published statuses.
owner_user_id	Information about the user that created the entity on ThreatStream.
organization_id	Information about the organization that created the entity on ThreatStream.
last_seen	Time stamp of when the malware was last observed, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
killchainstages	For internal use only.
workgroups	Workgroups with which the malware is shared.

Fields	Description
is_public	Whether the malware is public or private (including belonging to a trusted circle). False—if the malware is private or belongs to a Trusted Circle True—if the malware is public Default: False
circles	All Trusted Circles to which the malware belongs.
intelligence_ initiatives	Intelligence Initiatives associated with the malware.
assignee_user	The user to whom the malware is assigned.
first_seen	Time stamp of when the malware was first observed, in UTC format. Date must be specified in the following format: *YYYYMMDD*T*hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
is_anonymous	Whether the malware user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
feed_id	Numeric ID of the threat feed that provided the malware entity.
resource_uri	Internal URL for the malware within ThreatStream.
aliases	Other names by which the malware is known.

rules(limit)

This macro allows you to query statistics of the rules configured in ThreatStream. When used in Search, it returns the fields containing information related to rules, which then can be included in the operators for further analysis.

Fields	Description
create_ investigation	Whether an investigation was created the first time the rule was triggered.
created_ts	Time stamp of when the rule was created, in UTC format.
description	Description for the rule.
exclude_notify_ org_whitelisted	Whether Observables whitelisted by your organization are excluded from rule matches.
exclude_notify_ owner_org	Whether keyword matches on Observables imported by your organization are excluded from Keyword Match or Hourly Digest email notifications.
has_associations	Whether the rule creates associations between specified entities and matched entities.
id	Unique ID assigned to the rule
intelligence_ initiatives	Intelligence Initiatives associated with the rule.
is_editable	If the current user have edit access to the rule
is_enabled	If the rule is enabled.
keywords	A list of keywords associated with the rule.
match_actors	Unused.
match_all_tm	Unused.
match_ attackpatterns	Unused.
match_ campaigns	Unused.
mathc_ courseofactions	Unused.
match_ customtms	Unused.

Fields	Description
match_identities	Unused.
match_incidents	Unused.
match_ infrastructures	Unused.
match_ intrusionsets	Unused.
match_malware	Unused.
match_ observables	Whether the rule matches keywords in newly created Observables.
match_ reportedfiles	Whether the rule matches keywords in newly created Sandbox Reports.
match_signatures	Whether the rule matches keywords in newly created Signatures.
match_tips	Whether the rule matches keywords in newly created Threat Bulletins.
match_ttps	Unused.
match_ vulnerabilities	Whether the rule matches keywords in newly created Vulnerabilities.
matches	Total number of keyword matches for the rule.
modified_ts	Time stamp of when the rule was last modified, in UTC format.
name	Name of the rule.
org_id	ID associated with the organization that created the rule.
org_shared	Unused.
organization	ID, Name, and Resource URI associated with the organization that created the rule.

Fields	Description
resource_uri	Resource URI associated with the rule.
tags	Tags applied to matched entities.
user	User that created the rule.
user_id	ID of the user that created the rule.
workgroups	Workgroups associated with the rule.

signature(limit)

This macro allows you to query the data related to Signatures. When used in Search, it returns the fields containing Signature details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the Signature is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Signature.
circles	All trusted circles to which the Signature belongs.
created_ts	Time stamp of when the Signature was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
end_date	The time when this Signature is known to have ended. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Signature.
id	ID of the Signature. This ID is automatically assigned to the entity by ThreatStream at the time of creation.

Fields	Description
intelligence_initiatives	Intelligence Initiatives associated with the Signature.
is_anonymous	Indicates whether the Signature user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Signature can be cloned.
is_public	Indicates whether the Signature is public or private (including belonging to a trusted circle). False—if the Signature is private or belongs to a trusted circle True—if the Signature is public
modified_ts	Timestamp of when the Signature was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Signature. The Signature name is associated with your organization. Therefore, the name must be unique within your organization. However, two Signatures with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Signature on ThreatStream.
owner_user_id	ID of the user that created the Signature on ThreatStream.
publication_status	Signature entities can be in new, pending_review, review_requested, reviewed, or published statuses.

Fields	Description
published_ts	Timestamp of when the Signature was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Signature entity on ThreatStream.
s_type	Signature type.
source_created	Timestamp to when the Signature was first created.
source_modified	Timestamp to when the Signature was last updated.
tags	Tags assigned to the Signature. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Signature.
target_industry	Target industries associated with the Signature.
tlp	Traffic Light Protocol assigned to the Signature. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Signature for STIX compliance.
workgroups	Workgroups to which the Signature is visible.

tipreport(limit)

This macro allows you to query the data related to Threat Bulletins. When used in Search, it returns the fields containing Threat Bulletin details, which then can be included in the operators for further analysis.

Fields	Description
all_circles_visible	Indicates whether the Threat Bulletin is visible to all trusted circles.
assignee_org	Organization to which the Threat Bulletin is assigned
assignee_org_id	ID of the organization to which the Threat Bulletin is assigned.
assignee_org_name	Name of the organization to which the Threat Bulletin is assigned.
assignee_user	User to which the Threat Bulletin is assigned.
assignee_user_id	ID of the user to which the Threat Bulletin is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Threat Bulletin.
circles	All trusted circles to which the Threat Bulletin belongs.
created_ts	Time stamp of when the Threat Bulletin was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the Threat Bulletin.
id	ID of the Threat Bulletin. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Threat Bulletin.
is_anonymous	Indicates whether the Threat Bulletin user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Threat Bulletin can be cloned.

Fields	Description
is_email	Indicates whether the Threat Bulletin is email.
is_public	Indicates whether the Threat Bulletin is public or private (including belonging to a trusted circle). False—if the Threat Bulletin is private or belongs to a trusted circle True—if the Threat Bulletin is public
locations	Locations associated with the Threat Bulletin.
modified_ts	Timestamp of when the Threat Bulletin was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the Threat Bulletin. The Threat Bulletin name is associated with your organization. Therefore, the name must be unique within your organization. However, two Threat Bulletins with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
original_source	Original source from where the Threat Bulletin came.
original_source_id	ID of the original source of the Threat Bulletin.
owner_org	Organization owner of the Threat Bulletin.
owner_org_id	ID of the organization owning the Threat Bulletin.
owner_org_name	Name of the organization owning the Threat Bulletin.
parent	For internal use only.
publication_status	Signature entities can be in new, pending_review, review_requested, reviewed, or published statuses.

Fields	Description
published_ts	Timestamp of when the Threat Bulletin was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Threat Bulletin on ThreatStream.
source	Source of the Threat Bulletin intelligence.
source_created	Timestamp to when the Threat Bulletin was first created.
source_locations	Source locationa associated with the Threat Bulletin (as defined by STIX 2.1).
source_modified	Timestamp to when the Threat Bulletin was last updated.
status	Provides the status of the Threat Bulletin. Possible values—published, reviewed, review_ requested, pending_review.
tags	Tags assigned to the Threat Bulletin. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Threat Bulletin.
target_industry	Target industries associated with the Threat Bulletin.
threat_actor	Actors associated with the Threat Bulletin.
tlp	Traffic Light Protocol assigned to the Threat Bulletin. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the Threat Bulletin for STIX compliance.
workgroups	Workgroups to which the Threat Bulletin is visible.

ttp(limit)

This macro allows you to query the data related to TTPs. When used in Search, it returns the fields containing TTPs details, which then can be included in the operators for further analysis.

Fields	Description
assignee_user	ID of the user to which the TTP is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the TTP.
circles	All trusted circles to which the TTP belongs.
created_ts	Time stamp of when the TTP was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
feed_id	Numeric ID of the threat feed that provided the TTP.
id	ID of the TTP. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the TTP.
is_anonymous	Indicates whether the TTP user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_category	Indicates whether a category is assigned to TTP.
is_cloneable	Indicates whether the TTP can be cloned.
is_mitre	Indicates whether the TTP is described in the MITRE list. True—if the TTP is described False—if the TTP is not described

Fields	Description
is_public	Indicates whether the TTP is public or private (including belonging to a trusted circle). False—if the TTP is private or belongs to a trusted circle True—if the TTP is public
modified_ts	Timestamp of when the TTP was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
name	Name of the TTP. The TTP name is associated with your organization. Therefore, the name must be unique within your organization. However, two TTPs with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the TTP on ThreatStream.
publication_status	TTP entities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Timestamp of when the TTP was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the TTP entity on ThreatStream.
source_created	Timestamp to when the TTP was first created.
source_modified	Timestamp to when the TTP was last updated.

Fields	Description
tags	Tags assigned to the TTP. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the TTP.
target_industry	Target industries associated with the TTP.
tlp	Traffic Light Protocol assigned to the TTP. Possible values: red, amber, amber+strict, green, and clear.
uuid	UUID (universally unique identifier) assigned to the TTP for STIX compliance.
workgroups	Workgroups to which the TTP is visible.

vulnerability(limit)

This macro allows you to query the data related to Vulnerabilities. When used in Search, it returns the fields containing Vulnerability details, which then can be included in the operators for further analysis.

Fields	Description
aliases	Other names associated with the Vulnerability.
assignee_user	ID of the user to which the Vulnerability is assigned.
can_add_public_tags	Indicates whether a user can add public tags to the Vulnerability.
circles	All trusted circles to which the Vulnerability belongs.
created_ts	Time stamp of when the Vulnerability was created on ThreatStream, in UTC format. Date must be in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.

Fields	Description
feed_id	Numeric ID of the threat feed that provided the Vulnerability.
id	ID of the Vulnerability. This ID is automatically assigned to the entity by ThreatStream at the time of creation.
intelligence_initiatives	Intelligence Initiatives associated with the Vulnerability.
is_anonymous	Indicates whether the Vulnerability user and organization information is anonymized. True—if information is anonymized False—if information is not anonymized
is_cloneable	Indicates whether the Vulnerability can be cloned.
is_public	Indicates whether the Vulnerability is public or private (including belonging to a trusted circle). False—if the Vulnerability is private or belongs to a trusted circle True—if the Vulnerability is public
is_system	Indicates whether the Vulnerability was imported by ThreatStream. True—if the Vulnerability was imported by ThreatStream False—if the Vulnerability was added by a ThreatStream user
modified_ts	Timestamp of when the Vulnerability was last updated on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.

Fields	Description
name	Name of the Vulnerability. The Vulnerability name is associated with your organization. Therefore, the name must be unique within your organization. However, two Vulnerabilities with the same name can exist on ThreatStream as long as they belong to different organizations. The name can contain up to 255 characters.
organization_id	ID of the organization that created the Vulnerability on ThreatStream.
owner_user_id	ID of the user that created the Vulnerability on ThreatStream.
publication_status	Vulnerabilities can be in new, pending_review, review_requested, reviewed, or published statuses.
published_ts	Timestamp of when the Vulnerability was published on ThreatStream, in UTC format. Date is in this format: *YYYYMMDD***T***hhmmss* where T denotes the start of the value for time. For example, 2014-10-02T20:44:35.
resource_uri	URI of the Vulnerability entity on ThreatStream.
source	Source from where the Vulnerability was imported from.
source_created	Timestamp to when the Vulnerability was first created.
source_modified	Timestamp to when the Vulnerability was last updated.
tags	Tags assigned to the Vulnerability. A tag is a meaningful name or any other string value assigned to identify the information. For example, spear phishing, exploitation.
tags_v2	Additional information on the tags assigned to the Vulnerability.

Fields	Description
target_industry	Target industries associated with the Vulnerability.
tlp	Traffic Light Protocol assigned to the Vulnerability. Possible values: red, amber, amber+strict, green, and clear.
update_id	Updated Vulnerability ID.
uuid	UUID (universally unique identifier) assigned to the Vulnerability for STIX compliance.
workgroups	Workgroups to which the Vulnerability is visible.