

EclecticIQ Platform user guide

Ingest data with incoming feeds and enrichers — 1/4

Last generated: March 06, 2018



©2018 EclecticIQ

All rights reserved. No part of this document may be reproduced in any form or by any electronic or mechanical means, including information storage and retrieval systems, without written permission from the author, except in the case of a reviewer, who may quote brief passages embodied in critical articles or in a review.

Trademarked names appear throughout this book. Rather than use a trademark symbol with every occurrence of a trademarked name, names are used in an editorial fashion, with no intention of infringement of the respective owner's trademark.

The information in this book is distributed on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author nor the publisher shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this book.

©2018 by EclecticIQ BV. All rights reserved.
Last generated on Mar 6, 2018

Table of contents

Table of contents	2
User guide to EclecticIQ Platform	7
Scope	7
Goal	7
Audience	7
Feedback	7
About ingestion	9
Deduplication	9
Entity resolution	11
idref resolution at entity level	12
idref resolution at nested object level	13
Example of an empty placeholder entity	16
Saving data	17
Saving data URIs and raw artifacts	17
About ingestion discrepancies	18
Fewer entities in target platform	18
Entity count mismatch	19
About enrichment	20
Enrichment	20
Filtering and enriching	21
Saving data	22
Saving data URIs and raw artifacts	22
Configure incoming feeds	24
Configure the general options	25
Set a schedule	26
Set a TLP override	27
Set half-life values	27
Save options	28
Configure transport and content for specific incoming feeds	28
Start and stop feeds	30
Manually start a feed	30
Manually stop a feed	31
Suspend and disable a running feed	31
Stop and terminate a running feed	32
Delete and purge feeds	33
Delete an incoming feed	33
Purge an incoming feed	33
Configure IMAP email fetcher transport and content	35
About IMAP email fetcher	35
Configure transport and content types	35
Email body or attachment download known issues	37
Configure IMAP email attachment fetcher transport and content	38
About IMAP email attachment fetcher	38
Configure transport and content types	38
Email body or attachment download known issues	41
Configure FTP download transport and content	42
About FTP download	42
Configure transport and content types	42
Configure HTTP download transport and content	44
About HTTP download	44
Configure transport and content types	44
Configure mount point download transport and content	47
About Mount point download	47
Configure transport and content types	47

Configure SFTP download transport and content	50
About SFTP download	50
Configure transport and content types	50
Configure TAXII inbox transport and content	53
About TAXII inbox	53
Configure transport and content types	53
Push data to a TAXII inbox	54
Configure TAXII poll transport and content	57
About TAXII poll	57
Configure transport and content types	57
Configure Amazon S3 poll transport and content	61
About Amazon S3 poll	61
Configure transport and content types	61
Configure Anubis Cyberfeed transport and content	63
About Anubis Cyberfeed	63
Configure transport and content types	64
Configure BFK transport and content	66
About BFK API	66
Configure transport and content types	66
Configure Cisco Threat Grid Curated Feed transport and content	67
About Cisco Threat Grid Curated Feed	67
Configure transport and content types	68
Configure Cisco Threat Grid Samples API transport and content	69
About Cisco Threat Grid Samples API	69
Configure transport and content types	69
Configure CrowdStrike Falcon Intelligence Indicator transport and content	71
About CrowdStrike Falcon Intelligence Indicator	71
Configure transport and content types	71
Configure CrowdStrike Falcon Intelligence Reports transport and content	73
About CrowdStrike Falcon Intelligence Reports	73
Configure transport and content types	73
Configure CrowdStrike Falcon Intelligence Threat Actor transport and content	75
About CrowdStrike Falcon Intelligence Threat Actor	75
Configure transport and content types	75
Configure CVE Search API transport and content	77
About CVE Search API	77
Configure transport and content types	77
Configure FireEye iSIGHT Intelligence Report API transport and content	78
About FireEye iSIGHT Intelligence Report API	78
Configure transport and content types	78
Configure Group-IB transport and content	80
About Group-IB	80
Configure transport and content types	80
Configure Intel 471 IOC transport and content types	83
About Intel 471 IOC	83
Configure transport and content types	83
Configure Intel 471 Reports transport and content types	85
About Intel 471 Reports	85
Configure transport and content types	85
Configure MISP transport and content	87
About MISP API	87
Limitations	87
Configure transport and content types	88
About MISP processing	88
Resulting entities	88
Configure OpenPhish transport and content	91

About OpenPhish	91
Configure transport and content types	91
Configure PhishMe Intelligence transport and content	92
About PhishMe Intelligence	92
Configure transport and content types	92
Configure Proofpoint Email Brand Defense transport and content	94
About Proofpoint Email Brand Defense	94
Configure transport and content types	94
Configure SpyCloud Breach API transport and content	96
About SpyCloud Breach API	96
Configure transport and content types	96
Configure Threat Recon transport and content	97
About Threat Recon JSON API	97
Configure transport and content types	97
Incoming feeds reference	98
Available incoming feeds	98
Content types	100
Transport types	102
Configure enrichers	105
Configure the general options	105
Edit enricher tasks	106
Configure enricher rules	107
Add enricher rules	107
Save options	108
Edit enricher rules	108
Delete enricher rules	109
Configure specific enricher parameters	110
Enable and disable enrichers	112
View enricher tasks	112
Enable and disable enrichers	112
Run enrichers	114
Enrich entities	114
Enrich automatically	114
Enrich manually	115
Review enrichment observables	118
Review enrichment observables on the graph	118
Search for enrichment observables	121
Censys enricher	125
Configure the Censys enricher parameters	125
Query the Censys API	127
CIRCL IPs related to SSL certificate enricher	130
Configure the CIRCL IPs related to SSL certificate enricher parameters	130
CIRCL SSL Certificate Fetcher enricher	132
Configure the CIRCL SSL Certificate Fetcher enricher parameters	132
Cisco Threat Grid enricher	134
Configure the Cisco Threat Grid enricher parameters	134
CrowdStrike Falcon Intelligence Indicator enricher	136
Configure the CrowdStrike Falcon Intelligence Indicator enricher parameters	136
CVE Search enricher	138
Configure the CVE Search enricher parameters	138
DomainTools Hosted Domains enricher	139
Configure the DomainTools Hosted Domains enricher parameters	139
DomainTools Malicious Server Domains enricher	140
Configure the DomainTools Malicious Server Domains enricher parameters	140
DomainTools Reputation enricher	142
Configure the DomainTools Reputation enricher parameters	142

DomainTools Parsed Whois enricher	144
Configure the DomainTools Parsed Whois enricher parameters	144
DomainTools Suspicious Domains enricher	146
Configure the DomainTools Suspicious Domains enricher parameters	146
Elasticsearch sightings enricher	148
Configure the Elasticsearch sightings enricher parameters	148
Farsight DNSDB enricher	151
Configure the Farsight DNSDB enricher parameters	151
FireEye iSIGHT enricher	153
Configure the FireEye iSIGHT enricher parameters	153
Flashpoint AggregINT enricher	155
Configure the Flashpoint AggregINT enricher parameters	155
Flashpoint Blueprint enricher	157
Configure the Flashpoint Blueprint enricher parameters	157
Flashpoint Thresher enricher	159
Configure the Flashpoint Thresher enricher parameters	159
Fox-IT InTELL Portal enricher	161
Configure the Fox-IT InTELL Portal enricher parameters	161
Intel 471 enricher	163
Configure the Intel 471 enricher parameters	163
MISP API enricher	165
Limitations	165
Configure the MISP API enricher parameters	166
OpenDNS OpenResolve enricher	168
Configure the OpenDNS OpenResolve enricher parameters	168
PassiveTotal IP/Domain enricher	169
Configure the PassiveTotal IP/Domain enricher parameters	169
PassiveTotal Malware enricher	171
Configure the PassiveTotal Malware enricher parameters	171
PassiveTotal Passive DNS enricher	173
Configure the PassiveTotal Passive DNS enricher parameters	173
PassiveTotal Whois enricher	175
Configure the PassiveTotal Whois enricher parameters	175
Proofpoint Email Threat enricher	177
Configure the Proofpoint Email Threat enricher parameters	177
PyDat enricher	179
Configure the PyDat enricher parameters	179
Recorded Future enricher	181
Configure the Recorded Future enricher parameters	181
RIPEstat GeoIP enricher	183
Configure the RIPEstat GeoIP enricher parameters	183
RIPEstat Whois enricher	184
Configure the RIPEstat Whois enricher parameters	184
Shodan enricher	185
Configure the Shodan enricher parameters	185
Splunk sightings enricher	187
Configure the Splunk sightings enricher parameters	187
SpyCloud Breach Data enricher	190
Configure the SpyCloud Breach Data enricher parameters	190
ThreatCrowd enricher	192
Configure the ThreatCrowd enricher parameters	192
Unshorten-URL enricher	194
Configure the Unshorten-URL enricher parameters	194
VirusTotal enricher	195
Configure the VirusTotal enricher parameters	195
Create custom enrichers	197

About extensions	197
Create enricher extensions	197
Prepare the boilerplate	198
Edit the setup file	198
Include dependencies	199
Set the entry point	199
Define the enricher initialization	200
Build the enricher	201
Import dependencies	203
Set the UI schema definition	204
Define the UI schema	205
Field attributes	206
Define the enricher behavior	207
Define transport and content handlers	210
Define the data provider	210
Define the data transformer	212
Package and deploy the enricher	215
Restart processes	216
Check that the enricher is registered	216
Make an API call with HTTPie	216
API call response	218
Enable the enricher	218
Initialize the enricher	219
Create and run the fixtures	219
Test the enricher	219
Run extension validation	220
Error handling	220
Test the enricher with a test file	221
Test the enricher through the platform UI	223
Disable extensions	227
Enrichers reference	228
Enrichers	228
Default enrichers	228
Enricher types	231
Enricher input	235
Enricher output	237
Enrichment data types	240

User guide to EclecticIQ Platform

This user guide helps you configure the main options of the platform, as well as familiarize with EclecticIQ Platform, so that you can start collecting and analyzing potential threats efficiently.

Scope

The user guide to EclecticIQ Platform aims at providing clear and to-the-point help to get you acquainted with the threat intelligence platform, so that you can configure it as needed, and you can use it to collect and analyze intelligence on potential threats, as well as share it and collaborate with other analysts.

Although it is not a complete reference manual, this guide shows end-users how they can use the platform and its rich feature set to collect data, to analyze and investigate potential threats, and to collaborate and share intelligence with other analysts.

Goal

Learn how to incorporate the platform in your daily workflow as a powerful tool to:

- Automate data ingestion
- View, edit, create, and delete platform entities
- Enrich entities with additional contextual details
- Analyze entities on the graph to identify potential threats and their relationships
- Search, filter, and slice data using rules
- Share your findings and collaborate

Audience

This document targets the following audience:

- Cyber threat intelligence analysts
- Cyber threat intelligence specialists

Feedback

No one reads manuals, ever. We know.

Yet, we strive to give you clear, concise, and complete documentation that helps you get stuff done neatly.

We are committed to crafting good documentation, because life is too short for bad doc.

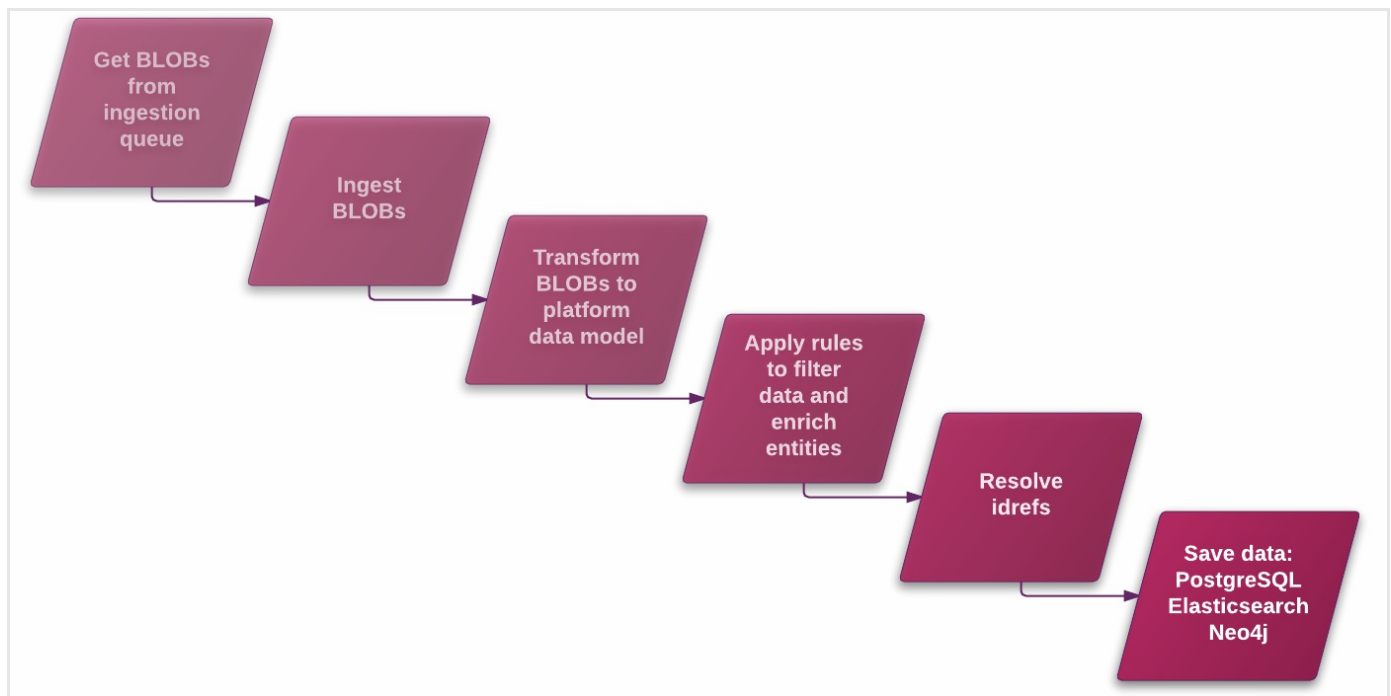
We appreciate your comments, and we'd love to hear from you: if you have questions or suggestions, drop us a line and share your thoughts with us!

👉 The Product Team

About ingestion

EclecticIQ Platform can ingest data in different formats from multiple sources. Before storing ingested data as entities and observables, the input data is deduplicated, normalized, and processes to search for, identify, and set any existing relationships.

Data ingestion into the platform is a multi-step process:



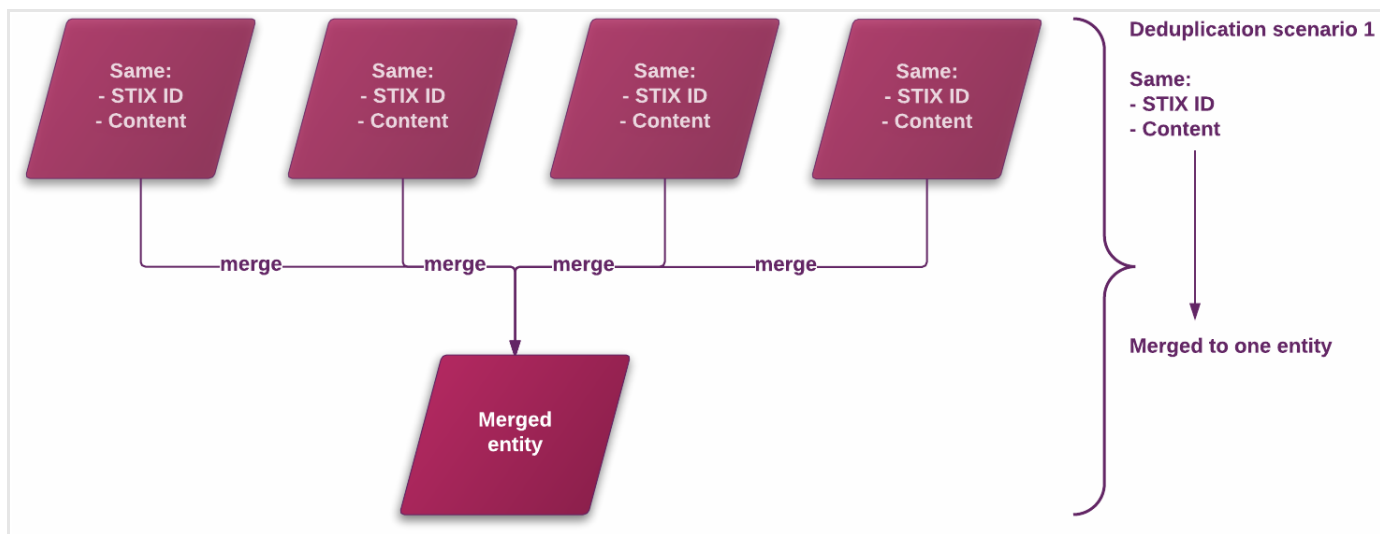
- Incoming data flows in and it is added to the ingestion queue.
- Blobs/Packages are fetched from the ingestion queue to be processed.
- Blobs/Packages are processed:
 - Data is deduplicated;
 - Data is normalized;
 - Data is transformed to the platform internal data model:
 - Entities
 - Observables
 - Relationships.
- Rules and filters enrich entities, create relationships, flag and tag entities, and so on.
- The platform tries to resolve ID references (`idrefs`) by looking for the data the IDs point to.
- The ingested entities are saved to the databases, they are indexed, they are searchable, and available for analysis.

Deduplication

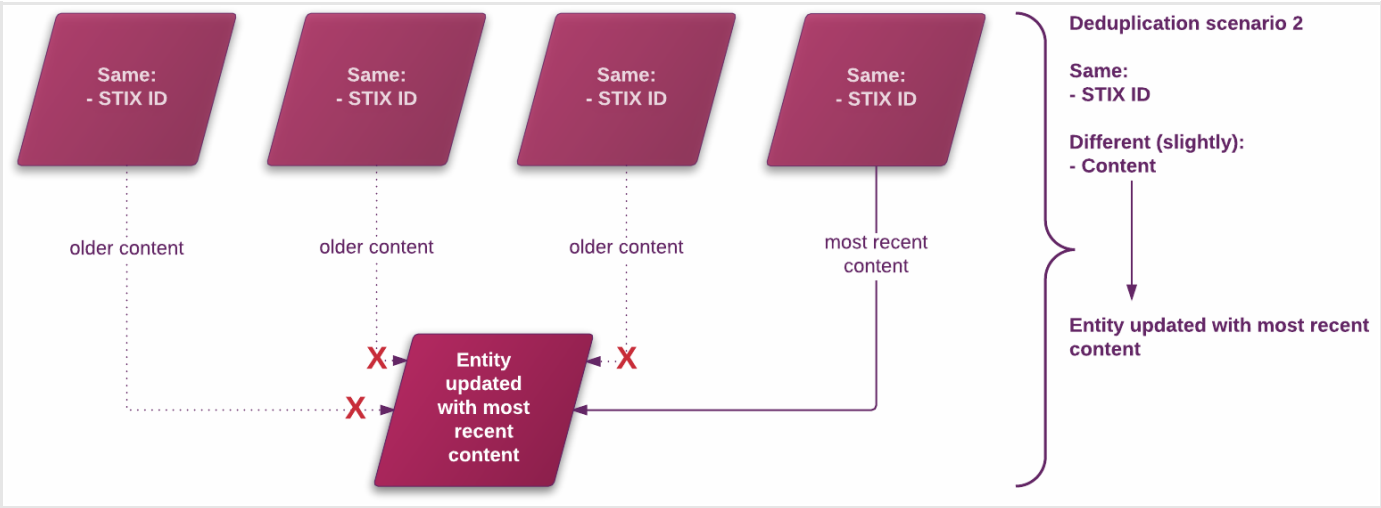
When checking newly ingested entities vs. existing ones, entity-level deduplication handles the following scenarios:

- Multiple entities sharing the *same STIX ID* and having *identical content* are handled like identical copies. In this case, identical copies of the same entity are merged to one entity.
- Multiple entities sharing the *same STIX ID* and having *slightly different content* are handled like versions of the same entity. In this context, *slightly different content* means different timestamp, and very minor differences in the entity content. For example, a reference version update, or a small edit such as fixing a typo in the title, description, or another similar field. In this case, the most recent content is used to update the existing entity. This avoids creating redundant copies of the same entity in the system.
- Multiple entities sharing the *same STIX ID* and having *identical content*, but *different timestamps* are handled like chronological versions of the same entity. In this case, the existing entity timestamp is updated to the most recent value without creating a new version of the entity.

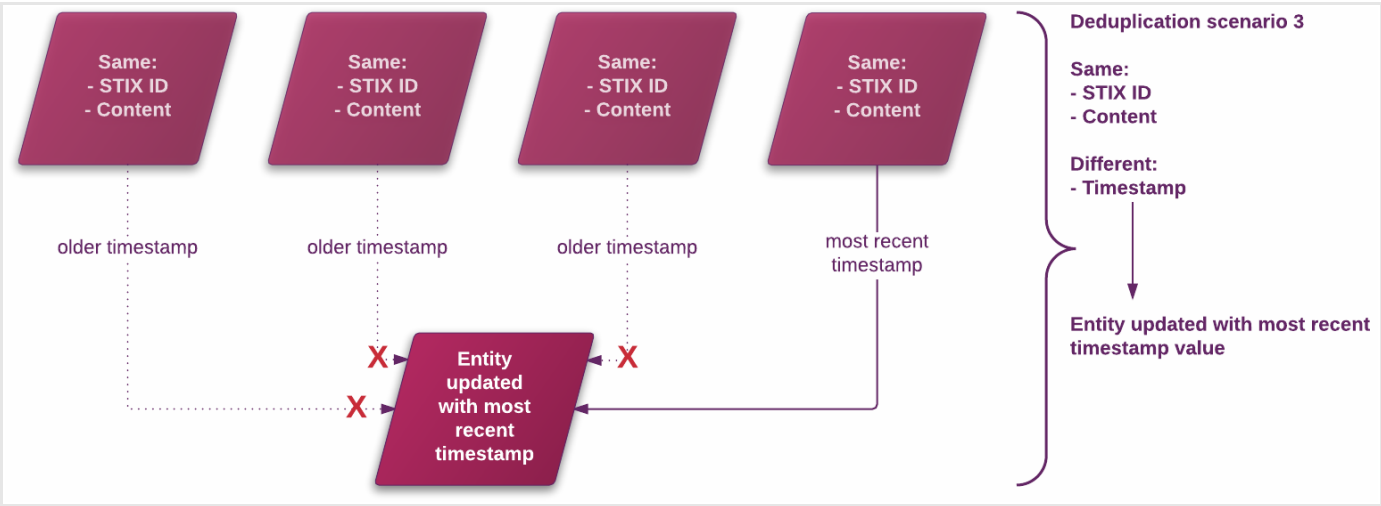
Before deduplication	After deduplication
Multiple entities	They are merged to one entity
Same STIX ID	
Identical content	
(identical copies)	



Before deduplication	After deduplication
Multiple entities	The existing entity is updated with the most recent content
Same STIX ID	
<i>Slightly different content</i>	
(versions of the same entity)	



Before deduplication	After deduplication
Multiple entities	The existing entity timestamp is updated to the most recent value
Same STIX ID	
Identical content	
<i>Different timestamps</i>	
(chronological versions of the same entity)	



Entity resolution

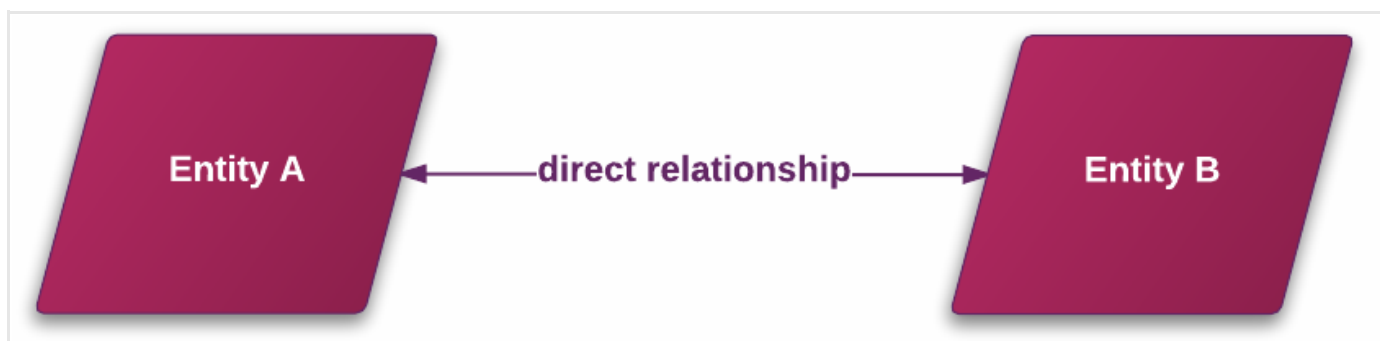
- An entity is *resolved* when: its `idref` is matched to the actual data it points to.
- An entity is *unresolved* when: its `idref` has no available matching data in the platform yet.

idref resolution at entity level

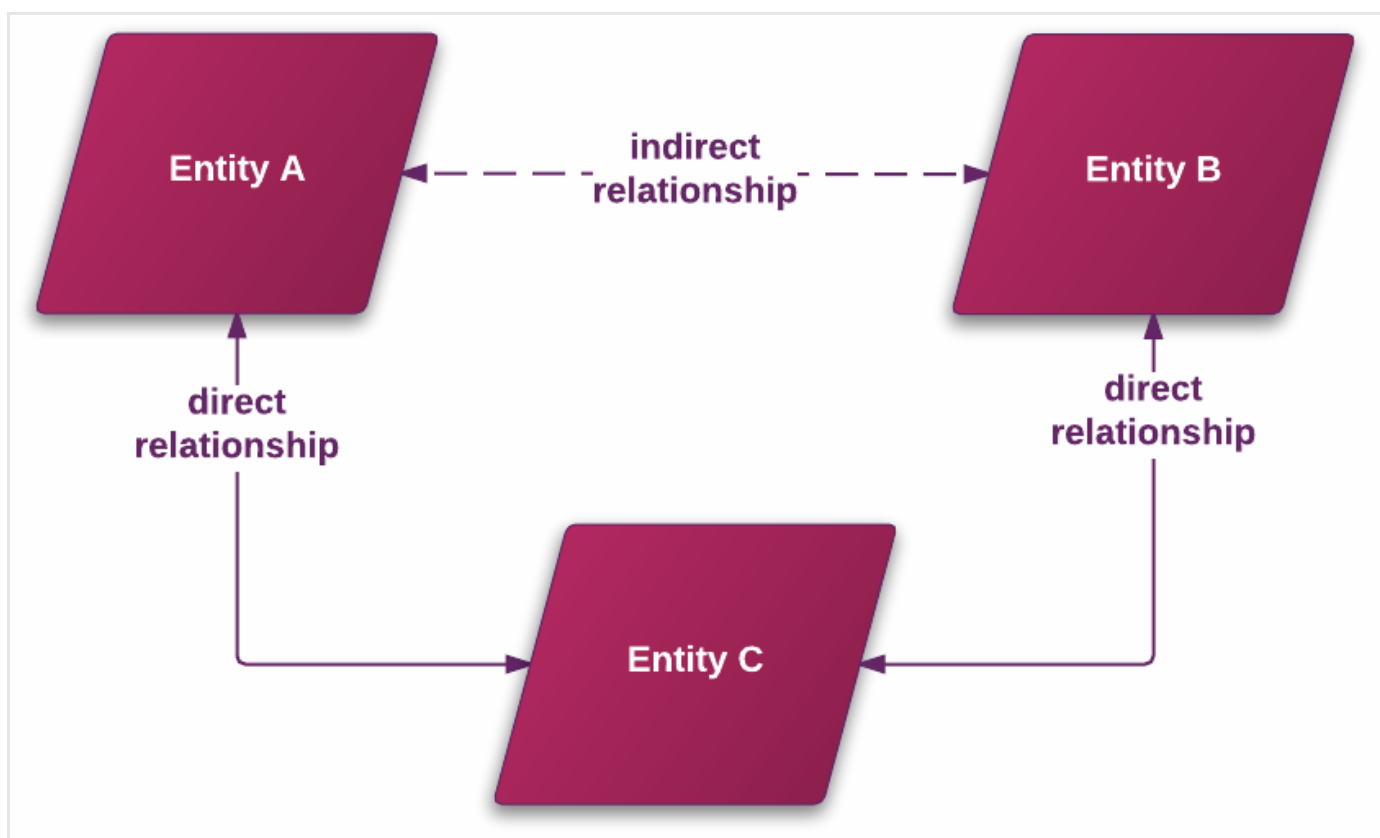
Entities and observables can reference external data through the STIX and CybOX `idref` field.

When the `idref` --> `idreference` is at entity level (STIX), the platform creates an entity relationship between the entities.

- When a STIX `idref` directly references an entity, the platform creates a direct relationship to associate the two entities.



- When a STIX `idref` indirectly references an entity, for example by establishing a relationship with the target entity through a connecting entity or an observable, the platform creates an indirect relationship to associate the two entities.



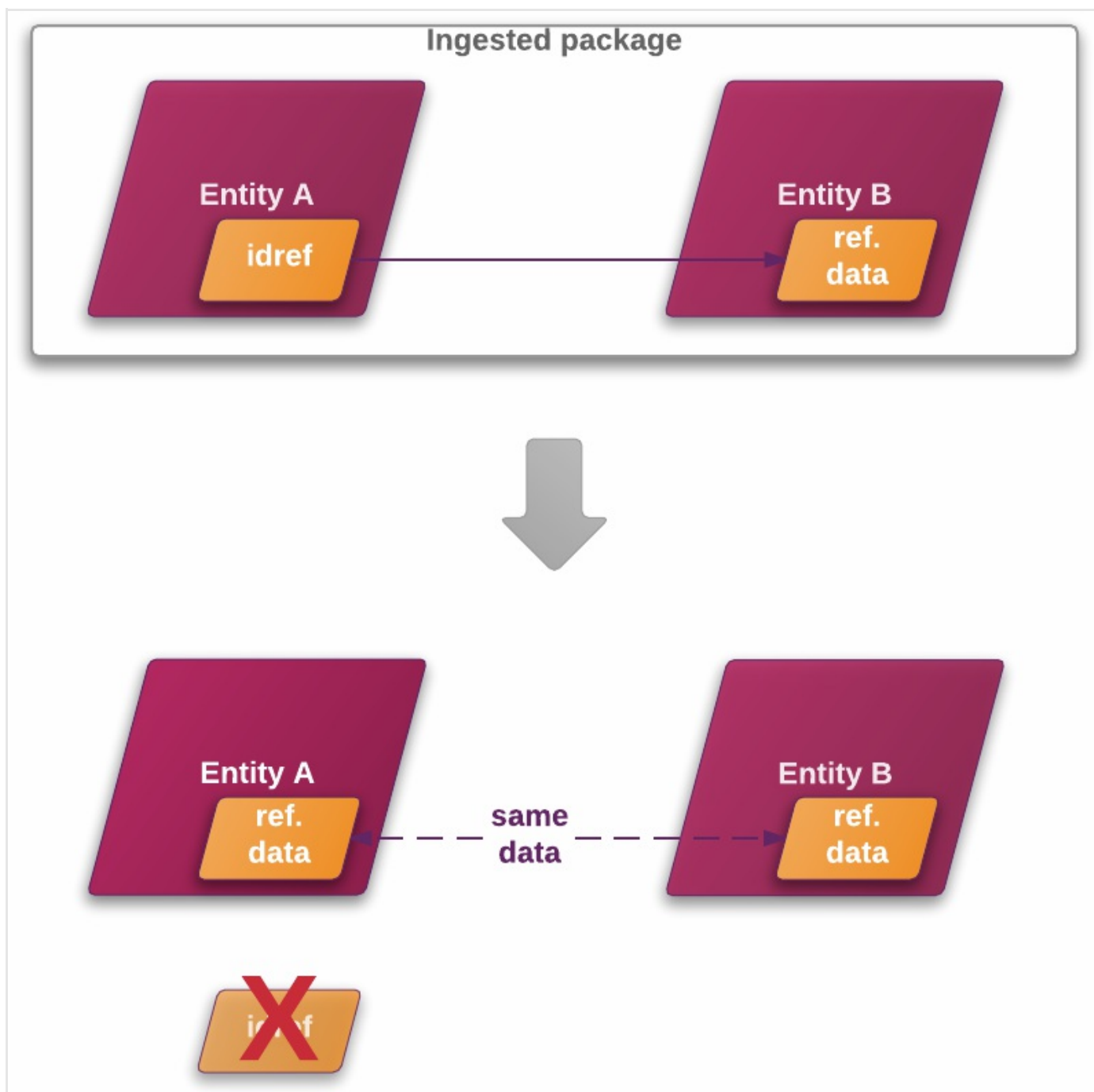
- When a STIX `idref` references an entity of the same type and content as the entity it belongs to, the only differences being either the `timestamp`, or the version reference values between the two entities, the platform processes the entity with the more recent timestamp or with the higher version value as an update of the other entity.



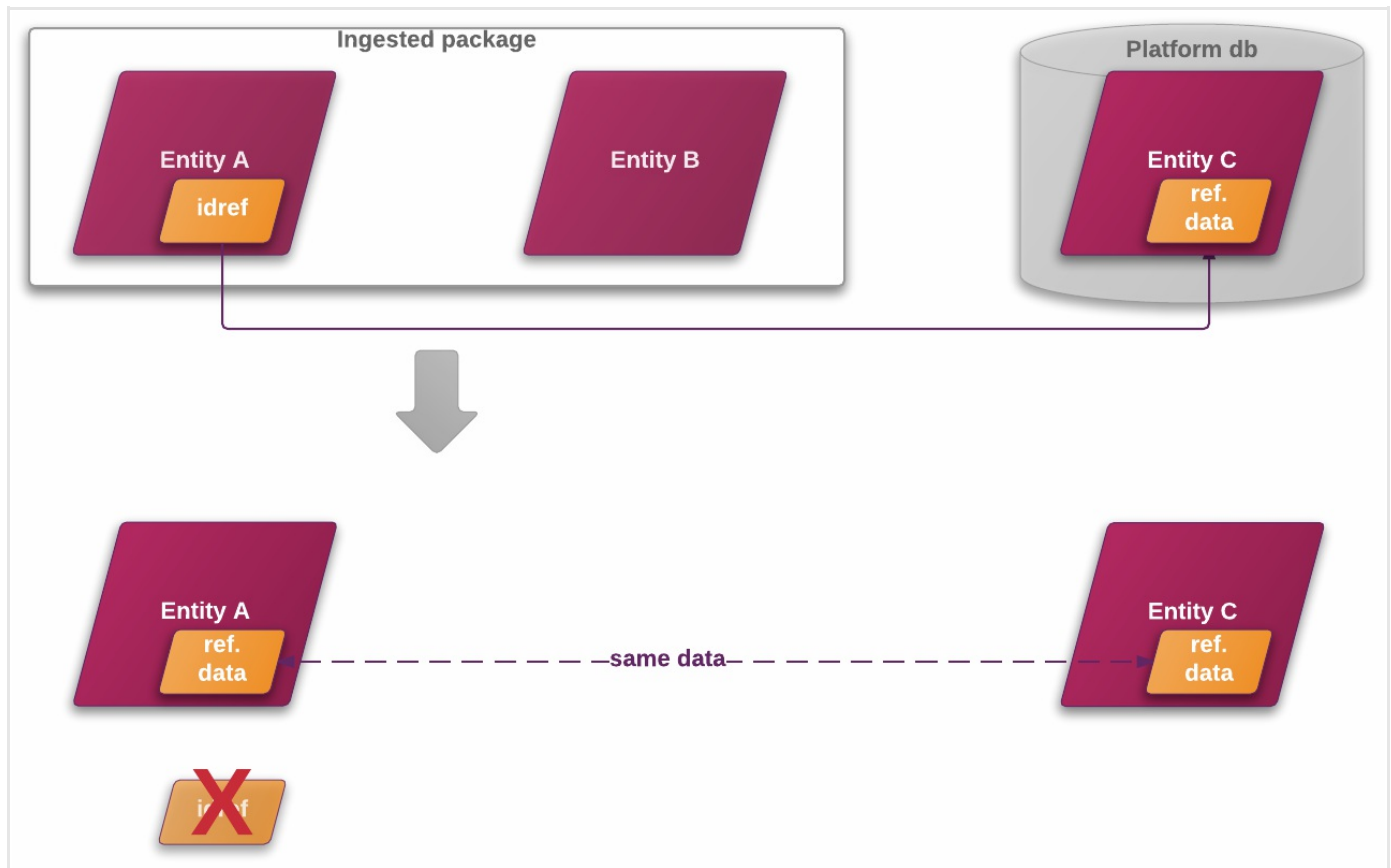
idref resolution at nested object level

When the `idref` --> `idreference` is at nested object level, that is, when an entity includes a bundled observable/an embedded CybOX observable object with an `idref`, the platform attempts to resolve the bundled object/CybOX `idref` by looking for the referenced data:

- If the bundled object/CybOX `idref` references data available inside the same ingested package or data stored in the platform database, the newly ingested `idref` is removed and it is replaced with the existing referenced data.

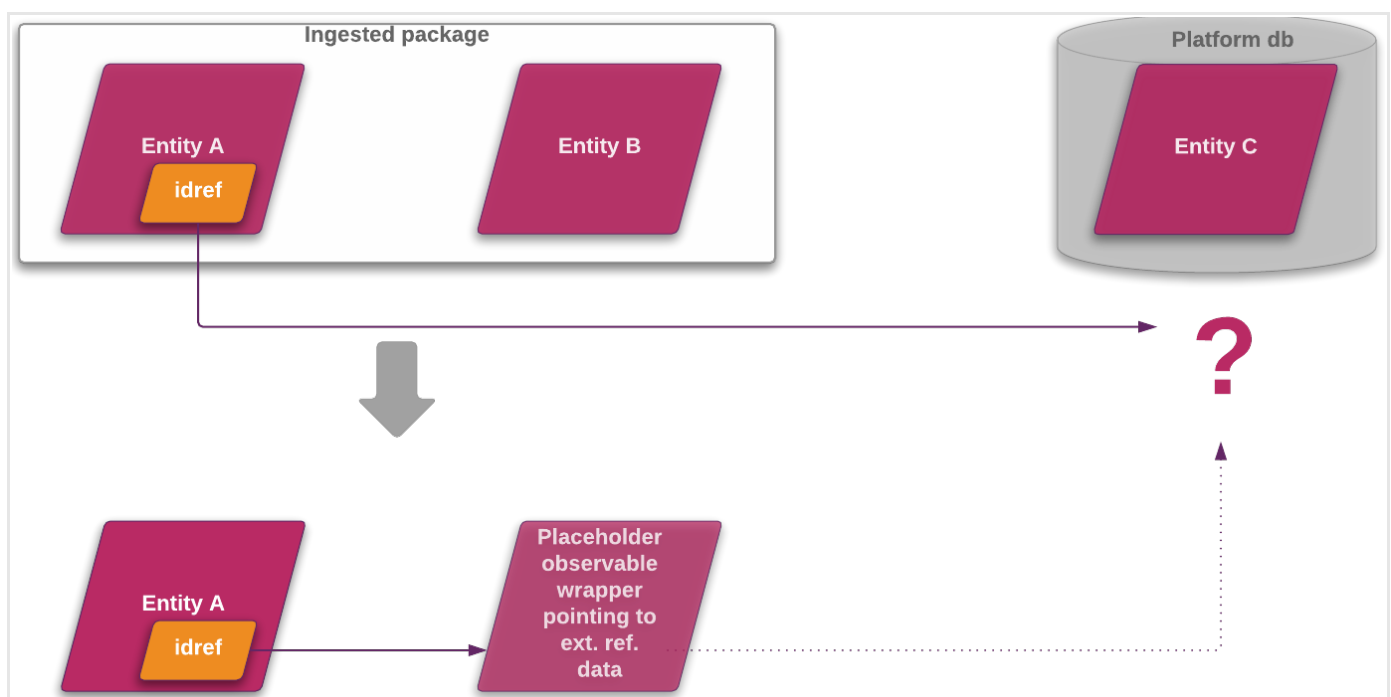


Referenced data is available inside the same ingested package



Referenced data is not available in the same ingested package, but available outside it in the platform database

- If the `idref` references unavailable data at the moment, the reference is resolved if/when the referenced data becomes available:
- The `idref` is converted to an *empty observable wrapper* containing only basic metadata such as `timestamp` and `idref`. It is populated with actual data when/if the originally referenced data becomes available.
- This is an empty placeholder object with no intel value, yet; therefore, it is not indexed and it is not searchable.



Referenced data is unavailable, an empty placeholder observable wrapper points to it, in case it becomes available in the future

The process works identically in the opposite direction: if bundled observable/CybOX data is ingested, the platform looks for an existing `idref` pointing to it.

- If it finds a matching `idref`, the existing `idref` is removed, and it is replaced with the newly ingested referenced data.
- If no matching `idref` is available, nothing happens.
If a matching `idref` becomes available at a later time, it will be resolved then by replacing it with the corresponding referenced data.

Example of an empty placeholder entity

This is an example of an empty TTP placeholder, as shown in the entity detail pane:

The screenshot displays the EclecticIQ interface. On the left, a list of entities is shown under the 'ENTITIES' tab. One entity, 'External reference to {http://www.fox-it.c...', is highlighted with a red box. On the right, the entity detail pane is open for this entity. The pane has a teal header with the title 'External reference to {http://www.fox-it.c...' and a close button. Below the header, it shows 'Ingested: 10/31/2016 5:19 PM' and 'Incoming feed: FoxIT2'. A 'TLP Amber' label is also present. The main content area has tabs for 'OVERVIEW', 'OBSERVABLES', 'NEIGHBORHOOD', 'JSON', 'VERSIONS', and 'HISTORY'. The 'OVERVIEW' tab is selected, and a red box highlights a warning message: 'THIS IS A PLACEHOLDER ENTITY. This placeholder exists because other entities refer to it. Its actual data is not (yet) available.' Below the warning, the 'Title' is shown as 'External reference to {http://www.fox-it.com/detact/ns/stix/1.0}attack-d1273cc8b3e706f2e0971a27694fdd1b'.

This is an example of an empty TTP placeholder in JSON format:

```
{
  "entities": [
    ...
    {
      "data": {
        "description": "fake batch aqtvfcfsfjzweljatukvdqunvkirmj",
        "id": "{https://fake.stix}ttp-aqtvfcfsfjzweljatukvdqunvkirmj-8481",
        "timestamp": "2018-02-02T14:18:27.970268+00:00",
        "title": "Fake TTP aqtvfcfsfjzw #8481",
        "type": "ttp"
      },
    },
    ...
  ]
}
```

Saving data

Last but not least, the platform saves the ingested entities to the databases in the following order:

- Entity store (PostgreSQL)
- Search store (Elasticsearch)
- Graph store (Neo4j)

Saving data URIs and raw artifacts

The extracted entity data that is stored inside observables ranges from short, simple data such as email addresses, domain names, IP addresses, and so on, to binary data. When an entity contains binary data — for example, a file, a memory region, or packet capture (PCAP) data — the data can be represented as either a *data URI* or a *CyBOX raw artifact* element.

During ingestion, extraction logic handles binary data URI and raw artifact objects embedded in CyBOX objects in the following way:

- **data URIs** (https://en.wikipedia.org/wiki/data_uri_scheme) are extracted and stored as entity attachments and new hash values:
 - The data URI value is recalculated to a new hash: `uri-hash-sha256`.
The SHA-256 hash value for `uri-hash-sha256` is calculated over the UTF-8 encoding of the data URI string.
The `uri-hash-sha256` hash substitute allows for entity correlation among entities containing the same data URI.
 - The binary data/raw content embedded in the data URI is decoded and processed:
 - The extracted binary data content is stored as an entity attachment similar to the `name="Raw_Artifact"` `type="ArtifactObj:RawArtifactType"` CyBOX object.
 - The extracted content is hashed using SHA-512, SHA-256, SHA-1, and MD5.
Each resulting hash is added to the relevant entities as an observable.

Example

A data URI with image content nested inside a CyBOX object generates the following output:

- 1 `uri-hash-sha256` hash to facilitate entity correlation
- 4 calculated hash observables: `hash-sha512`, `hash-sha256`, `hash-sha1`, and `hash-md5`
- 1 embedded JSON entity attachment (`raw-artifact`) with the extracted binary data representing the image content.

The following example shows a sample input along with the corresponding output.

```

dataUriExtractionSample(

  input={
    data:image/gif;base64,R0lGODlhAQABAAAAACH5BAEKAAEALAAAAABAAEAAAICTAEAOw==
  },

  output={

    # Recalculated hash of the original URI:
    ('uri-hash-sha256:'
     'd16ae5d51dda6f58995171aa23c0fa5e'
     '6dcd9c777cf9c251c4be3b1d62fdf670'),

    # Multiple hashes of the decoded content:
    'hash-md5:3eacd0132310ea44cad756b378a3bc07',

    'hash-sha1:e2216a7e9b73f5cb0279351c78ce61c33475cea7',

    ('hash-sha256:'
     'bb229a48bee31f5d54ca12dc9bd960c6'
     '3a671f0d4be86a054c1d324a44499d96'),

    ('hash-sha512:'
     'bd9ab35dde3a5242b04c159187732e13'
     'b0a6da50ddcff7015dfb78cdd68743e1'
     '91eaf5cddedd49bef7d2d5a642c21727'
     '2a40e5ba603fe24ca676a53f8c417c5d'),

    # (Attachment) Raw artifact as embedded JSON with the content:
    ('raw-artifact:{"content": '
     '"R0lGODlhAQABAAAAACH5BAEKAAEALAAAAABAAEAAAICTAEAOw==", '
     '"content_encoding": "base64", "type": "image/png"}'),

  }),

```

About ingestion discrepancies

Fewer entities in target platform

When you exchange entity data between two platform instances, and if the source data is in a format other than **EclecticIQ JSON**, an entity count discrepancy between the source and the destination platforms may arise at the end of the operation.

This is expected behavior.

For example, let's assume the source data format is STIX:

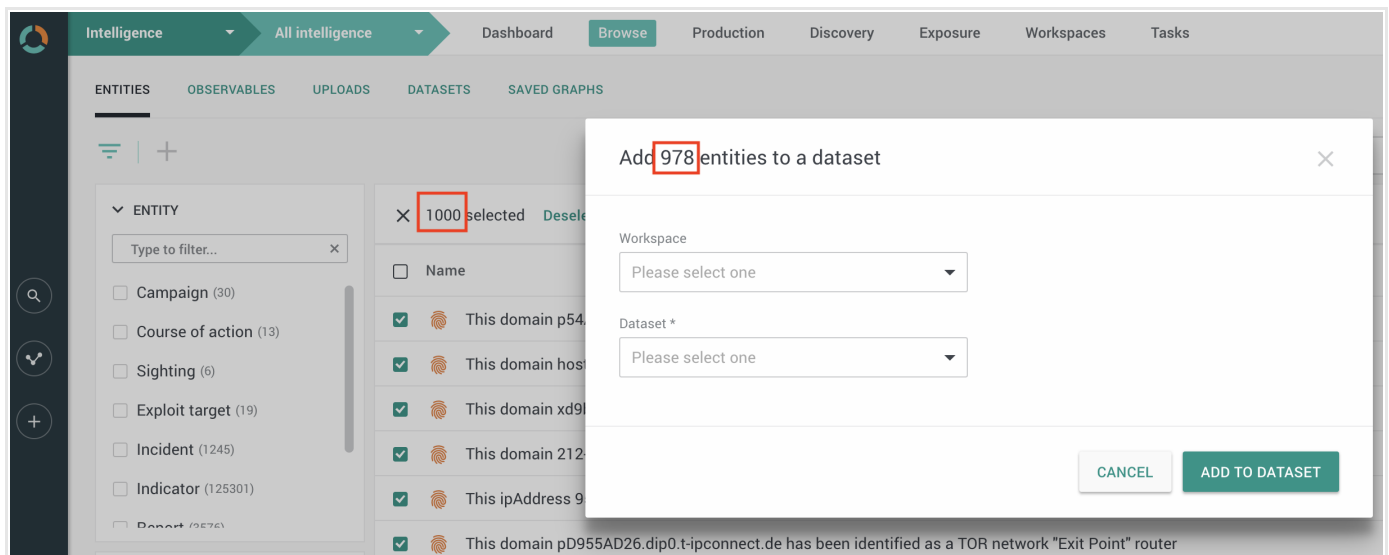
- STIX and JSON are different formats.
- STIX is much more nested than JSON, which has a flatter structure.
- STIX entities are bundled in packages; after ingestion, the package is not needed any longer.

- The ingestion process needs to parse the source data before ingesting it:
 - During parsing, STIX relationships between the package and the entities it contains are discarded, as they hold no intel value.
 - If the source package contains unresolved entities or observables whose `idref` is resolved during the parsing and ingestion steps, the original unresolved objects are discarded, as they hold no intel value because they are now resolved inside the platform.
 - Relationships between entities or between entities and observables are retained and ingested into the platform.

What looks like a discrepancy is the result of a data optimization and normalization process whose purpose is to separate the wheat from the chaff to retain only data with actual intelligence value.

Entity count mismatch

When you add a batch of entities to a dataset, the UI may display different entity counts on the dataset view and on the **Add [n] entities to a dataset** dialog window.



This UI inconsistency can impact all supported entity types.

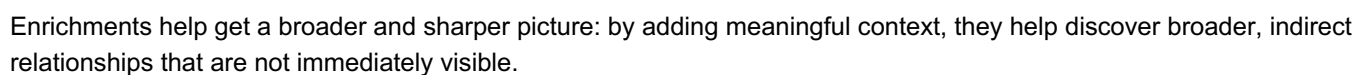
A possible cause for the count mismatch can be malformed STIX input, particularly malformed `idrefs` that can point to existing data — these `idrefs` are resolved when the platform replaces them with the matching data — or to data that is not yet available in the platform — these `idrefs` remain unresolved as long as there is no matching data to replace them with.

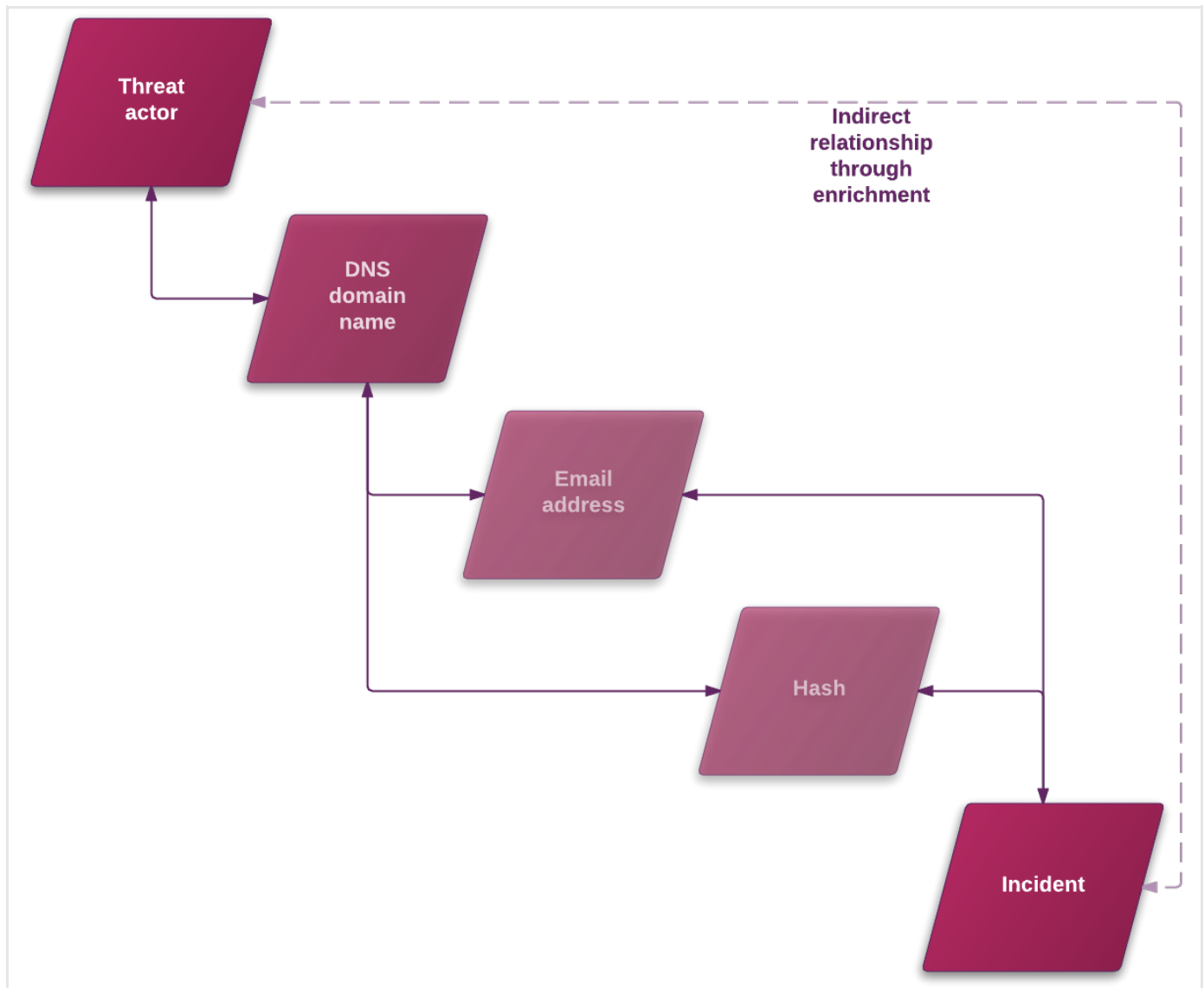
The count inconsistency is at UI level only, and it can occur in the following scenario:

- New data is being written to the platform, AND
- A user is examining the database view counter *and* the the **Add [n] entities to a dataset** dialog window counter at the same time, that is, both UI counters are displayed at the same time.

Enrichment augments existing cyber threat intelligence value by adding contextual information. Enrichers use rules and tasks to automatically enrich entities, so that you can explore a broader and more granular cyber threat intelligence landscape.

This process does not alter core entity data: each bit of enriching information is saved to observables, which are related to entities.





Enrichments augment observables with raw data information related to entities:



Filtering and enriching

Data extraction produces observables from data retrieved in embedded CybOX objects. This process contribute to data fusion across the whole platform dataset.

- Enrichment rules sift through data to augment it with enrichment observables, that is, observables obtained from the available enrichment data sources, and to exclude specific data based on the defined filtering rules.
- You can further refine the quality of the extracted data by applying ad-hoc rules to classify observables as safe or malicious.

STIX and CyBOX objects can both include placeholder references to external objects and data in the `idref` field. By default, the platform attempts to resolve `idrefs` at entity level and at nested object level by looking for the actual data the `idref` value represents and refers to. If it finds matching data, it creates either a relationship between entities — entity-level, STIX `idref` resolution — or it replaces the `idref` placeholder value with the corresponding actual data — nested object-level, CyBOX `idref` resolution.

Saving data

Last but not least, the platform saves the ingested entities to the databases in the following order:

- Entity store (PostgreSQL)
- Search store (Elasticsearch)
- Graph store (Neo4j)

Saving data URIs and raw artifacts

The extracted entity data that is stored inside observables ranges from short, simple data such as email addresses, domain names, IP addresses, and so on, to binary data. When an entity contains binary data — for example, a file, a memory region, or packet capture (PCAP) data — the data can be represented as either a *data URI* or a *CyBOX raw artifact* element.

During ingestion, extraction logic handles binary data URI and raw artifact objects embedded in CyBOX objects in the following way:

- **data URIs** (https://en.wikipedia.org/wiki/data_uri_scheme) are extracted and stored as entity attachments and new hash values:
 - The data URI value is recalculated to a new hash: `uri-hash-sha256`.
The SHA-256 hash value for `uri-hash-sha256` is calculated over the UTF-8 encoding of the data URI string.
The `uri-hash-sha256` hash substitute allows for entity correlation among entities containing the same data URI.
 - The binary data/raw content embedded in the data URI is decoded and processed:
 - The extracted binary data content is stored as an entity attachment similar to the `name="Raw_Artifact"` `type="ArtifactObj:RawArtifactType"` CyBOX object.
 - The extracted content is hashed using SHA-512, SHA-256, SHA-1, and MD5.
Each resulting hash is added to the relevant entities as an observable.

Example

A data URI with image content nested inside a CyBOX object generates the following output:

- 1 `uri-hash-sha256` hash to facilitate entity correlation
- 4 calculated hash observables: `hash-sha512`, `hash-sha256`, `hash-sha1`, and `hash-md5`
- 1 embedded JSON entity attachment (`raw-artifact`) with the extracted binary data representing the image content.

The following example shows a sample input along with the corresponding output.

```
dataUriExtractionSample(  
  
  input={  
    data:image/gif;base64,R0lGODlhAQABAAAAACH5BAEKAAEALAAAAABAAEAAAICTAEAOw==  
  },  
  
  output={  
  
    # Recalculated hash of the original URI:  
    ('uri-hash-sha256:'  
     'd16ae5d51dda6f58995171aa23c0fa5e'  
     '6dcd9c777cf9c251c4be3b1d62fdf670'),  
  
    # Multiple hashes of the decoded content:  
    'hash-md5:3eacd0132310ea44cad756b378a3bc07',  
  
    'hash-sha1:e2216a7e9b73f5cb0279351c78ce61c33475cea7',  
  
    ('hash-sha256:'  
     'bb229a48bee31f5d54ca12dc9bd960c6'  
     '3a671f0d4be86a054c1d324a44499d96'),  
  
    ('hash-sha512:'  
     'bd9ab35dde3a5242b04c159187732e13'  
     'b0a6da50ddcff7015dfb78cdd68743e1'  
     '91eaf5cddedd49bef7d2d5a642c21727'  
     '2a40e5ba603fe24ca676a53f8c417c5d'),  
  
    # (Attachment) Raw artifact as embedded JSON with the content:  
    ('raw-artifact:{"content": '  
     '"R0lGODlhAQABAAAAACH5BAEKAAEALAAAAABAAEAAAICTAEAOw==", '  
     '"content_encoding": "base64", "type": "image/png"}'),  
  }  
),
```


Configure incoming feeds

Configure incoming feeds to ingest data from selected sources in many different formats.

**Danger:**

Due to breaking changes, the following enrichers and incoming feeds are *not working* on EclecticIQ Platform release 1.14.4 (and earlier) until release 2.0.2 included:

- Enrichers:
 - FireEye iSIGHT
 - Intel 471
 - Recorded Future
- Incoming feeds:
 - BFK API
 - Cisco Threat Grid Curated Feed
 - Cisco Threat Grid Samples API
 - Crowdstrike Falcon Intelligence Indicator Feed
 - Crowdstrike Falcon Intelligence Reports Feed
 - Crowdstrike Falcon Intelligence Threat Actor Feed
 - FireEye iSIGHT Intelligence Report API

To restore full functionality for these enrichers and incoming feeds, upgrade to EclecticIQ Platform release 2.1.0 or later.

EclecticIQ Platform can ingest cyber threat intelligence made available in multiple formats. You can configure incoming feeds to retrieve cyber threat data from many different sources. Incoming feeds provide data, context, and additional information to help analysts draw an accurate map of the threat landscape they are examining. This knowledge is essential to making informed decisions quickly and efficiently, and to promoting a proactive behavior when monitoring IT infrastructures and managing IT security.

You can populate the platform with data by defining one or more incoming feed sources.

Once it is set up and it is running, an incoming feed provides a data stream that the platform ingests and processes automatically.

A minimal incoming feed configuration includes:

- A *data source*: the intel origin the incoming feed fetches data from.
For example, a URI, a path pointing to a network location, or an IP address to an API endpoint.
- A *transport type*: the vehicle carrying the data.
Typically, this is a communications protocol like TAXII, HTTP, FTP, or IMAP.
- A *content type*: the incoming data format the platform should expect from the incoming feed.
For example, STIX, JSON, CSV, or PDF.

About content

Structured content

For example: STIX, JSON, CSV

Machine-readable, easy to analyze, parse, and ingest as structured entities and observables.

Unstructured content

For example: PDF, email, (unstructured) text

Human-readable, more difficult to analyze, parse, and ingest as structured entities and observables.

When the platform ingests unstructured content — either through an incoming feed or a manual file upload — to produce a report entity, the original unstructured source is attached in its original format to the resulting report entity. In this way, when analysts modify or update the the resulting report entity, they can always refer back to the original information.

This article describes how to configure the **general options** for incoming feeds to ingest data into EclecticIQ Platform. These options are identical for all incoming feeds.

To configure transport type and content type, as well as any other specific options for a particular incoming feed, follow the links under Configure transport and content for specific incoming feeds .

Configure the general options



Input fields marked with an asterisk are required.

The **Incoming feeds** page displays an overview of the configured incoming feeds ingesting data from the specified intel providers and data sources.

- On the top navigation bar, select **Data configuration > Incoming feeds** .
- On the top-left corner of the page click the **+** icon to open the incoming feed editor.
- On the **Create incoming feed** form you can populate the input fields to define the intel provider/data source for the feed, and the feed behavior.
- Under **Feed name**, enter a name for the feed you are creating. It should be descriptive and easy to remember.
- Under **Organization**, enter a name for the organization that serves as the intel provider for the incoming feed.
- Use **Source reliability** to flag the incoming feed with a value from the drop-down list to help oher users assess how trustworthy the feed source is deemed to be.

This value has the same meaning as the first character in the **two-character Admiralty System code**

(https://en.wikipedia.org/wiki/admiralty_code).

- **Require valid signature:** select this checkbox to enforce PGP signature validation of the incoming feed data.

To work correctly, the source data provider needs to offer a PGP validation service based on a public and a private key.

Moreover, you need to obtain a valid PGP public key from the source data provider, which you need to add to the platform trusted keys.

To add a trusted key to the platform, do the following::

- On the left-hand navigation sidebar click **⚙ > System settings > Trusted keys > Edit settings** .
- On the **Edit public trusted keys settings** page, click **+ Add** or **+ More** to add a new PGP public key:
 - **Key:** copy-paste into this field the public PGP key you want to add as a trusted key to verify the PGP signature of incoming data packages.
Include in the pasted content the leading `-----BEGIN PGP PUBLIC KEY BLOCK-----` and the trailing `-----END PGP PUBLIC KEY BLOCK-----` lines.
 - **Description:** add a short description to help users understand the purpose of the PGP public key.
 - Click **Save** to store your changes, or **Cancel** to discard them.
- **Skip extraction of observables from unstructured text:** select this checkbox to exclude from ingestion observable data detected in unstructured content and without link names defining their relationships, if any.

If you select the checkbox, the platform filters out any observable data detected inside titles, headers, descriptions, summaries, and other free-form, free text fields.
Observable data inside unstructured text fields is usually not as relevant, and not as valuable in terms of intelligence, as observable data extracted from, for example, CybOX fields.
In the same way, observables with relationships, but without any link names providing extra context and relevance, can add more noise than actual value to the platform data.
- From the **Transport type** and **Content type** drop-down menus select the appropriate options to configure transport and content for the specified incoming feed.

Set a schedule

Under **Schedule — Execution schedule** you can define how often you want to automatically run the feed task:

- **None:** scheduled feed execution is disabled. You need to manually trigger the task to ingest or to publish data through an incoming or an outgoing feed, respectively.
- **Every [n] minutes:** the feed task runs automatically once every [n] minutes, where [n] defines the selected time interval in minutes.
You define the execution interval in 5-minute increments from the corresponding drop-down menu.
- **Every hour, [n] minutes past the hour:** the feed task runs automatically once an hour every hour at the specified minute offset from the hour.
You define how long in minutes after the beginning of an hour the task should run from the corresponding drop-down menu.
- **Every [n] hours:** the feed task runs automatically once every [n] hours, where [n] defines the time interval in hours between two consecutive feed task runs.
You define how long the time interval between feed executions should be by selecting the number of hours from the corresponding drop-down menu.
- **Every day at [time]:** the feed task runs automatically once a day at the specified time.
You define the time of the day when the task should run from the corresponding drop-down menus.

- **Every [n] days**: the feed task runs automatically once every [n] days, where [n] defines the time interval in days between two consecutive feed task runs.
You define how long the time interval between feed executions should be by selecting the number of days from the corresponding drop-down menu.
- **Every week on [day of the week] at [time]**: the feed task runs automatically once a week on the designated day, at the specified time.
You define the day of the week and time of the day when the task should run from the corresponding drop-down menus.
- **Every month on [day of the month] at [time]**: the feed task runs automatically once a month on the designated day of the month, at the specified time.
You define the day of the week and time of the day when the task should run from the corresponding drop-down menus.
Keep in mind that not all months of the year have 30 or 31 days.

Set a TLP override

Override TLP overwrites the **TLP** (<https://www.us-cert.gov/tlp>) color code associated with the feed entities with the one you set here. The selected TLP value is assigned to all the entities in the feed.

You can override the original or the current TLP color code of an entity, an incoming feed, or an outgoing feed.

When working as a filter, TLP colors select a decreasing range: if you set a TLP color as a filter the enricher, the feed, or the returned filtered results include all the entities flagged with the selected TLP color code, as well as all the entities whose TLP color indicates that they are progressively lower risk, less sensitive, and suitable for disclosure to broader audiences.

For example, if you select green the filtered results include entities with a TLP color set to green, as well as entities with a TLP color set to white, and entities with no TLP color code flag.

Set half-life values

It represents the amount of time it takes an entity to lose half its intelligence value.

It corresponds to the number of days it takes the intelligence value of a malicious entity to decay by 50%.

When configuring an incoming or an outgoing feed, you can set a half-life value in days for the following entity types:

- **Campaign**
- **Course of action**
- **Exploit target**
- **Incident**
- **Indicator**
- **TTP**
- **Threat actor**
- **Report**

To set a half-life for one or more of these properties, do the following::

- Enter a numerical value in the entity property input field(s) you want to flag with a half-life value in days.

- Click **Save** to store your changes, or **Cancel** to discard them.

Save options

Besides committing the current data by clicking **Save**, you can also click the downward-pointing arrow on the **Save** button to display a context menu with additional save options:

- **Save and run**: saves the current configuration for the feed, and it executes is right away; that is, the feed task to ingest (incoming feeds) or disseminate (outgoing feeds) intelligence is triggered upon saving the feed configuration.
- **Save and new**: saves the current configuration for the feed, and it opens a new empty form to start configuring a new feed.
- **Save and duplicate**: saves saves the current configuration for the feed, and it opens a pre-populated copy of the same feed configuration, which you can use as a template to speed up manual work.

Configure transport and content for specific incoming feeds

- Configure Amazon S3 poll transport and content
- Configure Anubis Cyberfeed transport and content
- Configure BFK transport and content
- Configure CrowdStrike Falcon Intelligence Indicator transport and content
- Configure CrowdStrike Falcon Intelligence Reports transport and content
- Configure CrowdStrike Falcon Intelligence Threat Actor transport and content
- Configure CVE Search API transport and content
- Configure IMAP email fetcher transport and content
- Configure IMAP email attachment fetcher transport and content
- Configure FireEye iSIGHT Intelligence Report API transport and content
- Configure FTP download transport and content
- Configure Group-IB transport and content
- Configure HTTP download transport and content
- Configure Intel 471 Reports transport and content types
- Configure Intel 471 IOC transport and content types
- Configure MISP transport and content
- Configure mount point download transport and content
- Configure NSFocus transport and content
- Configure OpenPhish transport and content
- Configure PhishMe Intelligence transport and content
- Configure Proofpoint Email Brand Defense transport and content

- Configure SFTP download transport and content
- Configure SpyCloud Breach API transport and content
- Configure TAXII inbox transport and content
- Configure TAXII poll transport and content
- Configure Cisco Threat Grid Samples API transport and content
- Configure Cisco Threat Grid Curated Feed transport and content
- Configure Threat Recon transport and content
- Exchange data between platforms
- About ingestion

Start and stop feeds


Enable and disable feeds, as well as manually trigger a feed task run or stop a running feed.

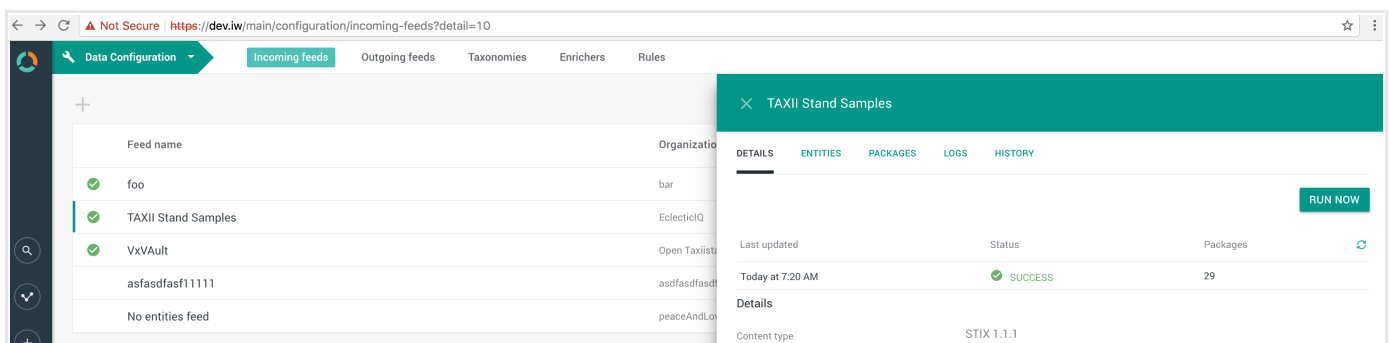
After configuring a feed, you can set a schedule to automate feed execution over time. If you do not set an execution schedule, the feed does not run, that is, it does not fetch or publish any data.

Manually start a feed

You can manually start an incoming or an outgoing feed run in one of the following ways:

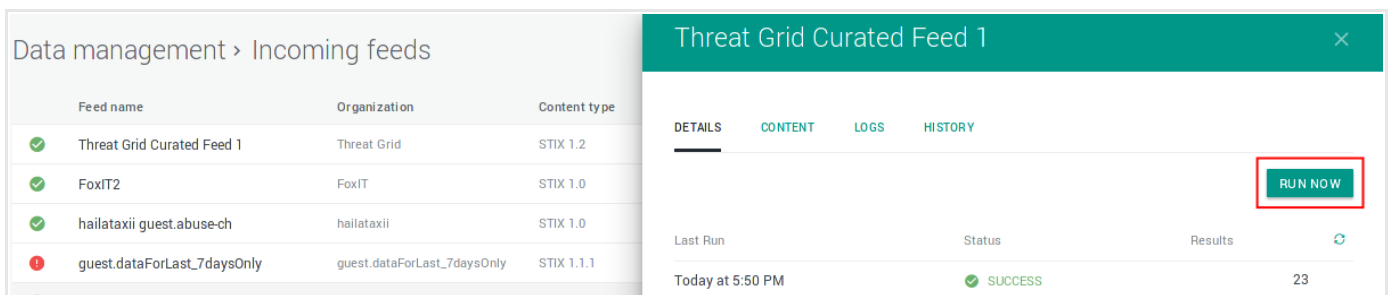
On the incoming feed overview page

- Go to **Data configuration > Incoming feeds** or to **Data configuration > Outgoing feeds**, depending on whether you want to run an incoming or an outgoing feed.
- On the feed overview page, click the  icon corresponding to the feed you want to run.
- From the drop-down menu select **Run now**.



On the incoming feed entity detail pane

- Go to **Data configuration > Incoming feeds** or to **Data configuration > Outgoing feeds**, depending on whether you want to run an incoming or an outgoing feed.
- On the feed overview page, click anywhere on the row corresponding to the feed you want to run.
- On the **Details** tab on feed detail pane click **Run now**.



Through the Actions menu

- Go to **Data configuration > Incoming feeds** or to **Data configuration > Outgoing feeds**, depending on whether you want to run an incoming or an outgoing feed.
- On the feed overview page, click anywhere on the row corresponding to the feed you want to run.

- On the **Details** tab on feed detail pane, scroll to the bottom of the pane, and then click **Actions**.
- From the pop-up menu select **Run now**.

Threat Grid Curated Feed 1 ×

DETAILS **CONTENT** **LOGS** **HISTORY**

RUN NOW

Last Run	Status	Results	
Today at 5:50 PM	✓ SUCCESS	23	↻

Details

Content Type	STIX 1.2
Transport Type	Cisco AMP Threat Grid Curated Feed
Organisation	Threat Grid
Execution Schedule	Every day at 00:00
Override TLP	None
Extraction Ignore Levels	2

Transport Configuration

API URL	https://panacea.threatgrid.com/api/v3/
API Key	*****
	DNS Entries observed from samples signed with a stolen certificate
	2017-01-22T00:00:00+00:00

Run now

Edit

Delete

Actions

Manually stop a feed


You can either manually suspend/disable or stop/terminate a running incoming or outgoing feed.

Suspend and disable a running feed

To disable an active feed, do the following:

- Go to **Data configuration > Incoming feeds** or to **Data configuration > Outgoing feeds**, depending on whether you want to run an incoming or an outgoing feed.
- On the feed overview page, click anywhere on the row corresponding to the feed you want to run.
- On the **Details** tab on feed detail pane click **Disable**.
You can enable the disabled feed at any time by clicking **Enable**.

You can disable feed execution also by setting the feed execution schedule to **None**:



- Go to **Data configuration > Incoming feeds** or to **Data configuration > Outgoing feeds**, depending on whether you want to suspend an incoming or an outgoing feed.
- On the feed overview page, click the  icon corresponding to the feed you want to run.
- From the drop-down menu select **Edit**.
- On the feed configuration page, go to the **Schedule** section, and then set **Execution schedule** to **None**.
- Click **Save** to store your changes, or **Cancel** to discard them.

Alternatively:

- Go to **Data configuration > Incoming feeds** or to **Data configuration > Outgoing feeds**, depending on whether you want to run an incoming or an outgoing feed.
- On the feed overview page, click anywhere on the row corresponding to the feed you want to run.
- On the feed detail pane, scroll to the bottom of the pane, and then click **Actions**.
- From the drop-down menu select **Edit**.
- On the feed configuration page, go to the **Schedule** section, and then set **Execution schedule** to **None**.
- Click **Save** to store your changes, or **Cancel** to discard them.

Stop and terminate a running feed

To stop the execution of a running feed and kill the task, do the following:

- On the left-hand navigation sidebar, click  > **System jobs > Running**.
- On the **System jobs > Running** overview page, browse to the running task(s) you want to terminate, and then click the corresponding  **Terminate** button to instantly stop executing the selected task(s).

Delete and purge feeds

If you do not need to keep an incoming feeds any longer, you can decide whether you want to delete the feed while keeping entities ingested through it, or purge the feed to delete it along with any linked data such as entities and relationships.

You may want to delete an incoming feed to avoid polluting the platform with unnecessary clutter or dirty data. For example, the content an incoming feed delivers may be no longer necessary, or an error in the feed configuration may have produced information you do not need.

In these cases, you can either delete or purge an incoming feed:

- *Delete the feed* to remove the incoming feed configuration.
 - The platform stops ingesting and processing data from the designated data source for the feed.
 - Existing data linked to the feed are preserved, such as previously ingested and processed entities and relationships.
- *Purge the feed* to remove the incoming feed configuration, and any data ingested through or linked to the feed.
 - The platform stops ingesting and processing data from the designated data source for the feed.
 - Data linked to the feed are completely removed as well, such as entities previously ingested through the feed.

Delete an incoming feed


Choose this option to:

- Stop ingesting data through the selected incoming feed.
- Delete the feed and its configuration from the platform.
- Keep the data ingested through the feed, such as entities and relationships.

To delete an incoming feed, do the following:

- On the top navigation bar click **Data configuration > Incoming feeds**.
- On the feed overview page click anywhere on the row corresponding to the feed you want to delete.
- On the feed detail pane, click **Actions > Delete**.

Alternatively:

- On the feed overview page click the  icon on the row corresponding to the feed you want to delete.
- From the drop-down menu select **Delete**.
- On the confirmation dialog, click **Delete** to confirm the action.

Purge an incoming feed

Choose this option to:


- Stop ingesting data through the selected incoming feed.

- Delete the feed and its configuration from the platform.
- Delete also any data ingested through the feed, such as entities and relationships.

To purge an incoming feed, do the following:

- On the top navigation bar click **Data configuration > Incoming feeds**.
- On the feed overview page click anywhere on the row corresponding to the feed you want to purge.
- On the feed detail pane, click **Actions > Delete**.

Alternatively:

- On the feed overview page click the  icon on the row corresponding to the feed you want to purge.
- From the drop-down menu select **Delete**.
- On the confirmation dialog, click **Purge** to confirm the action.

Configure IMAP email fetcher transport and content

Set up and configure transport and content types for IMAP email fetcher incoming feeds to retrieve and process information from email body contents.

To configure the general options for the IMAP email fetcher incoming feed, see [Configure incoming feeds](#).

About IMAP email fetcher

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Email message	Email message body content.	Structured, STIX-compliant report entities.

Configure transport and content types

Transport type	Allowed content types
IMAP email fetcher	Email message

The IMAP email fetcher transport type for incoming feeds fetches and ingests email body content in the supported content types.

Any existing email attachments are not processed, and they are deleted after processing and ingesting the body content of the corresponding email(s).

- From the **Transport type** drop-down menu, select **IMAP email fetcher**.
- From the **Content type** drop-down menu, select **Email message**.
The **IMAP email fetcher** transport type supports only the **Email message** content type.

Under **Transport configuration**, configure the following settings:

- **Host**: the address of the email server acting as a data source for the incoming feed.
Example: *imap.call-cthulhu.com*
- **Username**: a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Password**: a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Use SSL**: select this checkbox if the data source email server supports SSL, and if you want to enable the security layer for the feed.
- **To keyword**: you can enter a keyword to filter and include in the feed only content found in emails whose *To* recipient field includes the specified keyword search string.
- **From keyword**: you can enter a keyword to filter and include in the feed only content found in emails whose *From* sender field includes the specified keyword search string.

- **Subject keyword:** you can enter a keyword to filter and include in the feed only content found in emails whose *Subject* field includes the specified keyword search string.
- **SSL certificate authentication:** if the IMAP email server requires an SSL certificate to authenticate and to authorize access to email services, select this checkbox to fill out the required information.
- **SSL certificate:** copy-paste the content of a valid SSL certificate to authenticate.
SSL certificate file format: PEM
Example:

```
-----BEGIN CERTIFICATE REQUEST-----
MIICvDCCAaQCAQAwdzELMAkGA1UEBhMCVVMxDTALBgNVBAgMBFV0YWgxDzANBgNV
BACMBkxpbmRvbWVjEWMBQGA1UECgwNRGlnaUNlcnQgSW5jLjERMA8GA1UECwwIRGln
aUNlcnQxHTAbBgNVBAMMFjY4YUw1bWUuZGlnaWNlcnQuY29tMIIBIjANBgkqhkiG
9w0BAQEFAAOCAQ8AMIIBCgKCAQEa8+To7d+2kPWeBv/orU3LVbJwDrSQbeKamCmo
wp5bqDxIwV20zqRb7APUOKYoVEFFOEQs6T6gImnIolhbiH6m4zgZ/CPvWBOKZc+c
1Po2EmvBz+AD5sBdT5kzGQA6NbWyZGldxRthNLOslefOhdnWFuhI162qmcflgpiI
WDuwq4C9f+YkeJhNn9dF5+owm8cOQmDrV8NNdiTqin8q3qYAHHJRW28glJUCZkTZ
wIaSR6crBQ8TbYNE0dc+Caa3DOI kz1EOsHWzTx+n0zKfqcBgXi4DJx+C1bjptYPR
BPZL8DAeWuA8ebudVT44yEp82G96/Ggcf7F33xMxe0yc+Xa6owIDAQABoAAwDQYJ
KoZiHvcNAQEFAADgEBAAB0kcrFccSmFDmxox0Ne01UIqSsDqHgL+XmHTXJwre6D
hJSZwbvEtOKOG3+dr4Fs11WuUNT5qcLsx5a8uk4G6AKHMzuhLsJ7XZjgmQXGECpY
Q4mC3yT3ZocGPiXbw+iP3lmeEXgaQL0Tx5LF1/okKbKYWiQNiYKWOMj7ZR/wxWg/
ZDGRs55xuoelDJ/ZRFf9bI+IaCUd1YrfYcHl13G87Av+r49YVwqRDT0VDV7uLgqn
29XI1PpVUNCPQGN9p/eX6Qo7vpDaPybRtA2R7XLKjQaF9oXWeCUqylhvJac9QFO2
97Ob1alPpOZ7mWiEuJwJBpIi6a9M9G30nUo39lBilw=
-----END CERTIFICATE REQUEST-----
```

- **SSL key:** copy-paste the content of a valid SSL key to authenticate.
SSL key file format: PEM
Example:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEATz2mr7SZiAMfQyuvBjM9Oi..Z1BjP5CE/Wm/Rr500P
RK+Lh9x5eJPo5CAZ3/ANBE0sTK0ZsDGMak2mlg7..3VHqIxFTz0Ta1d+NAj
wnLe4nOb7/eEJbDPk05ShhBrJGBKKxb8n104o/.PdzbFMIYnJzBM2o5y
5A13wiLiteO7nco2WfyYkQzaxCw0AwzlkVHiIyC..71pSzkv6sv+4IDMbT/
XpCo8L6wTarzrywnQsh+etLD6FtTjYbbrvZ8RQM..Hg2qxraAV++HNBmNW
kbJ+q+rsJxQlaipn2M4lGuQJEfIxELFDyd3XpxP..Un/82NZNXlPmRIopXs
2T91jiLZEUKQw+n73j26adTbteuEaPGSrTzxBLR..yss00wWomUyILqVeti
+PK+aXKwguI6bxLGZ3of0UH+mGsS10mkp7kyZCm..OTQtfeRqP8rDSC7DgA
kHc5ajYqh04AzNFaxjRo+M3IGICUaOdKnXd0Fda..QwfoaX4QlRTgLqb7AN
ZTzM9WbmnYoXrx17kZlT3lsCgYEAm757XI3WJVj..WoLj1+v48WyoXZpcai
uv9bT4Cj+lXRS+gdKHK+SH7J3x2CRHVS+WH/SVC..DxuybvebDoT0TkKiCj
BWQaGzCaJqZa+POHK0klvS+9ln0/6k539p95tfX..X4TCzbVG6+gJiX0ysz
Yfehn5MCgYEAKMiKuWHCsVyCab3RUf6XA9gd3qY..fCTIGtS1tr5PgFIV+G
engiVoWc/hkj8SBHzz1n1xLN7KdF8ySU06MDggB..hJ+gXJKy+gf3mF5KmJ
DtkpjGHQzPF6vOe907y5NQLvVFGXUq/FIJZxB8k..fJdHEm2M4=
-----END RSA PRIVATE KEY-----
```

**Warning:**

Email body or attachment download known issues

Currently, the **IMAP email fetcher** and the **IMAP email attachment fetcher** incoming feed transport types do not support retrieving email body content or email attachments, respectively, from the following web-based email service providers:

- Microsoft Office 365 Outlook
- Google Gmail

These providers will be supported in the coming future.

Configure IMAP email attachment fetcher transport and content

Set up and configure transport and content types for IMAP email attachment fetcher incoming feeds to retrieve email attachment content in multiple formats. You can then use the resulting reports for analysis or automation purposes.

To configure the general options for the IMAP email attachment fetcher incoming feed, see [Configure incoming feeds](#).

About IMAP email attachment fetcher

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
IMAP email attachment fetcher	Structured and unstructured data in JSON, PDF, STIX, and plain text format.	Structured, STIX-compliant report entities.

Configure transport and content types

Transport type	Allowed content types
IMAP email attachment fetcher	CAPEC XML
	EclecticlQ JSON
	EclecticlQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

The IMAP email attachment fetcher transport type for incoming feeds fetches and ingests email attachments in the supported content types.

The email body is not processed, and the email message is deleted after processing and ingesting the corresponding attachment(s).

- From the **Transport type** drop-down menu, select **IMAP email attachment fetcher**.
- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.

The selected content type for the feed should match the data source format.

This can vary, depending on the intel sources you retrieve the data from.

The **IMAP email attachment fetcher** transport type enables fetching data in the following formats:

- *CAPEC XML*
- *EclecticIQ JSON*
- *EclecticIQ JSON (Legacy)*
- *PDF*
- *STIX 1.0*
- *STIX 1.1*
- *STIX 1.1.1*
- *STIX 1.2*
- *Text*
- **Accept password protected archives:** select this checkbox to specify a global password to open any archives retrieved through the incoming feed.

If the archives are password-protected, enter it in the **Archive password** field. The specified password acts as a master password, and it is used to try to unlock and access any archives retrieved with the feed.

Supported archive formats: *.rar, .tar, .tar.bz2, .tar.gz, .tar.bz2, .tar.z, .zip*

Under **Transport configuration**, configure the following settings:

- **Host:** the address of the email server acting as a data source for the incoming feed.
Example: *imap.call-cthulhu.com*
- **Username:** a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Password:** a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Use SSL:** select this checkbox if the data source email server supports SSL, and if you want to enable the security layer for the feed.
- **To keyword:** you can enter a keyword to filter and include in the feed only content found in emails whose *To* recipient field includes the specified keyword search string.
- **From keyword:** you can enter a keyword to filter and include in the feed only content found in emails whose *From* sender field includes the specified keyword search string.
- **Subject keyword:** you can enter a keyword to filter and include in the feed only content found in emails whose *Subject* field includes the specified keyword search string.

- **SSL certificate authentication:** if the IMAP email server requires an SSL certificate to authenticate and to authorize access to email services, select this checkbox to fill out the required information.
- **SSL certificate:** copy-paste the content of a valid SSL certificate to authenticate.
SSL certificate file format: PEM
Example:

```
-----BEGIN CERTIFICATE REQUEST-----
MIICvDCCAAQCAQAwdzELMAkGA1UEBhMCVVMxDTALBgNVBAgMBFV0YWgxDzANBgNV
BACMBkxpbmRvbWJEWMBQGA1UECgwNRGlnaUNlcnQgSW5jLjERMA8GA1UECwwIRGln
aUNlcnQxHTAbBgNVBAMMFjY4YUw1bGUuZGlnaWNlcnQuY29tMIIBIjANBgkqhkiG
9w0BAQEFAAOCAQ8AMIIBCgKCAQEAA8+To7d+2kPWeBv/orU3LVbJwDrSQbeKamCmo
wp5bqDxIwV20zqRb7APUOKYoVEFFOEQs6T6gImnIo1hbiH6m4zgZ/CPvWBOkZc+c
1Po2EmvBz+AD5sBdT5kzGQA6NbWyZGldxRthNLOslefOhdnWFuhI162qmcflgpiI
Wduwq4C9f+YkeJhNn9dF5+owm8cOQmDrV8NNdiTqin8q3qYAHHJRW28glJUCZkTZ
wIaSR6crBQ8TbYNE0dc+Caa3DOIkz1EOsHWzTx+n0zKfqcBgXi4DJx+C1bjptYPR
BPZL8DAeWuA8ebudVT44yEp82G96/Ggcf7F33xMxe0yc+Xa6owIDAQABoAAwDQYJ
KoZThvcNAQEFBQADggEBAB0kcrFccSmFDmxox0Ne01UIqSsDqHgL+XmHTXJwre6D
hJSZwbvEtOKG3+dr4Fs11WuUNT5qcLsx5a8uk4G6AKHMzuhLsJ7XZjgmQXGECpY
Q4mC3yT3Z0CGpIXbw+iP3lmeEXgaQL0Tx5LF1/okKbKYWiQNiYKWOMj7ZR/wxWg/
ZDGRs55xuoeLDJ/ZRff9bI+IaCud1YrfYcHil3G87Av+r49YVwqRDT0VDV7uLgqn
29XI1PpVUNCPQgn9p/eX6Qo7vpDaPybRtA2R7XLKjQaF9oXWeCUqylhvJac9QF02
970blalPpPoZ7mWiEuJwJBpIi6a9M9G30nUo391Bilw=
-----END CERTIFICATE REQUEST-----
```

- **SSL key:** copy-paste the content of a valid SSL key to authenticate.
SSL key file format: PEM
Example:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAA3Tz2mr7SZiAMfQyuvBjM9Oi..Z1BjP5CE/Wm/Rr500P
RK+Lh9x5eJPo5CAZ3/ANBE0sTK0ZsDGMak2m1g7..3VHqIXFTz0Ta1d+NAj
wnLe4nOb7/eEJbDPkk05ShhBrJGBKKxb8n104o/..PdzbFMIyNjJzBM2o5y
5A13wiLitEO7nco2WfyYkQzaxCw0AwzlkVHiIyC..71pSzkv6sv+4IDMbT/
XpCo8L6wTarzrywnQsh+etLD6FtTjYbbrvZ8RQM..Hg2qxraAV++HNBmNW
kbJ+q+rsJxQlaipn2M4lGuQJEfIXELFDyd3XpxP..Un/82NZNX1PmRIopXs
2T91jiLZEUKQw+n73j26adTbteuEaPGSrTZxBLR..yss00wWomUyILqVeti
+PK+aXKwguI6bxLGZ3of0UH+mGsS10mkp7kYZCm..OTQtfeRqP8rDSC7DgA
kHc5ajYqh04AZNFaxjRo+M3IGICUaOdKnXd0Fda..QwfoaX4QLRTgLqb7AN
ZTzM9WbmnYoXrx17kZlT3lsCgYEA757XI3WJVj..WoLj1+v48WyoXZpcai
uv9bT4Cj+1XRS+gdKHK+SH7J3x2CRHVS+WH/SVC..DxuybvebDoT0TkKiCj
BWQaGzCaJqZa+POHK0klvS+9ln0/6k539p95tfX..X4TCzbVG6+gJiX0ysZ
Yfehn5MCgYEAkMiKuWHCsVyCab3RUF6XA9gd3qY..fCTIGtS1tR5PgFIV+G
engiVoWc/hk78SBHzzln1xLN7KdF8ySU06MDggB..hJ+gXJKy+gf3mF5KmJ
DtKpjGHQzPF6v0E907y5NQLvVFGXuq/FIJZxB8k..fJdHEm2M4=
-----END RSA PRIVATE KEY-----
```

**Warning:**

Email body or attachment download known issues

Currently, the **IMAP email fetcher** and the **IMAP email attachment fetcher** incoming feed transport types do not support retrieving email body content or email attachments, respectively, from the following web-based email service providers:

- Microsoft Office 365 Outlook
- Google Gmail

These providers will be supported in the coming future.

Configure FTP download transport and content

Set up and configure transport and content types for FTP download incoming feeds to retrieve and process information from specific data sources supporting the FTP protocol.

To configure the general options for the FTP download incoming feed, see [Configure incoming feeds](#).

About FTP download

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
FTP download	Structured and unstructured data in JSON, PDF, STIX, and plain text format.	Structured, STIX-compliant entities and observables.

Configure transport and content types

Transport type	Allowed content types
FTP download	CAPEC XML
	EclecticIQ JSON
	EclecticIQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

- From the **Transport type** drop-down menu, select **FTP download**.

- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.
The selected content type for the feed should match the data source format.
This can vary, depending on the intel sources you retrieve the data from.
The **FTP download** transport type enables fetching data in the following formats:

- *CAPEC XML*
- *EclecticIQ JSON*
- *EclecticIQ JSON (Legacy)*
- *PDF*
- *STIX 1.0*
- *STIX 1.1*
- *STIX 1.1.1*
- *STIX 1.2*
- *Text*
- **Accept password protected archives:** select this checkbox to specify a global password to open any archives retrieved through the incoming feed.

If the archives are password-protected, enter it in the **Archive password** field. The specified password acts as a master password, and it is used to try to unlock and access any archives retrieved with the feed.

Supported archive formats: *.rar*, *.tar*, *.tar.bz2*, *.tar.gz*, *.tar.bz2*, *.tar.z*, *.zip*

Under **Transport configuration**, configure the following settings:

- **URL:** the location on the FTP server where the data source for the feed is made available.
- **Username:** a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Password:** a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **TLS:** select this checkbox if the data source FTP server supports TLS, and if you want to enable the security layer for the feed.

Configure HTTP download transport and content

Set up and configure transport and content types for HTTP download incoming feeds to retrieve and process information from specific data sources supporting the HTTP protocol.

To configure the general options for the HTTP download incoming feed, see [Configure incoming feeds](#).

About HTTP download

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
HTTP download	Structured and unstructured data in JSON, PDF, STIX, and plain text format.	Structured, STIX-compliant entities and observables.

Configure transport and content types

Transport type	Allowed content types
HTTP download	CAPEC XML
	EclecticIQ JSON
	EclecticIQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

- From the **Transport type** drop-down menu, select **HTTP download**.

- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.
The selected content type for the feed should match the data source format.
This can vary, depending on the intel sources you retrieve the data from.
The **HTTP download** transport type enables fetching data in the following formats:

- *CAPEC XML*
- *EclecticIQ JSON*
- *EclecticIQ JSON (Legacy)*
- *PDF*
- *STIX 1.0*
- *STIX 1.1*
- *STIX 1.1.1*
- *STIX 1.2*
- *Text*

- **Accept password protected archives:** select this checkbox to specify a global password to open any archives retrieved through the incoming feed.

If the archives are password-protected, enter it in the **Archive password** field. The specified password acts as a master password, and it is used to try to unlock and access any archives retrieved with the feed.

Supported archive formats: *.rar, .tar, .tar.bz2, .tar.gz, .tar.bz2, .tar.z, .zip*

Under **Transport configuration**, configure the following settings:

- **URL:** the location/directory/folder on the server or network unit hosting the data source for the feed.
- **Regex pattern:** you can define a regex to include in the incoming feed only the content delivered through the links that match the specified regex pattern.

Your input needs to be a **valid regex pattern** (<https://remram44.github.io/regex-cheatsheet/regex.html>).

Examples of valid regex patterns:

- *+.json*
- *+.pdf*
- ***
- *[^/]all_files_with_this_name_but_different_extensions[^\].[a-z0-9]**

- If you do not enter any regex pattern, the feed fetches the base URL response body, and it assumes that any retrieved content is in the format specified under **Content type**.
- If you enter a regex pattern, the feed assumes that any retrieved content is in HTML format, regardless of the format specified under **Content type**.
Retrieved content is scanned for any links matching the regex pattern.
Matches are downloaded, assuming that their format corresponds to the one specified under **Content type**.
- **Basic auth username:** a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
HTTP download uses basic HTTP authentication.
- **Basic auth password:** a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.

- **Verify connection:** select this checkbox if you want to test the connection to verify that it works as expected.
- **Extra headers:** the HTTP server hosting the data source for the feed may require passing additional HTTP headers in the request.
Click **+ Add** or **+ More** to include in the request one or more additional HTTP headers, along with the corresponding values:
 - *Accept-Charset*
 - *Accept-Encoding*
 - *Accept-Language*
 - *Accept-Datetime*
 - *Authorization*
 - *Connection*
 - *Cookie*
 - *Expect*
 - *From*
 - *Host*
 - *If-Match*
 - *If-Modified-Since*
 - *If-None-Match*
 - *If-Range*
 - *If-Unmodified-Since*
 - *Max-Forwards*
 - *Origin*
 - *Proxy-Authorization*
 - *User-Agent*
 - *Via*
 - *Warning*

Configure mount point download transport and content

Set up and configure transport and content types for mount point download incoming feeds to retrieve and process information from specific data sources available on the (local) network.

To configure the general options for the Mount point download incoming feed, see [Configure incoming feeds](#).

About Mount point download

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Mount point download	Structured and unstructured data in JSON, PDF, STIX, and plain text format.	Structured, STIX-compliant entities and observables.

Configure transport and content types

Transport type	Allowed content types
Mount point download	CAPEC XML
	EclecticlQ JSON
	EclecticlQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

- From the **Transport type** drop-down menu, select **Mount point download**.

- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.
The selected content type for the feed should match the data source format.
This can vary, depending on the intel sources you retrieve the data from.
The **Mount point download** transport type enables fetching data in the following formats:

- *CAPEC XML*
- *EclecticIQ JSON*
- *EclecticIQ JSON (Legacy)*
- *PDF*
- *STIX 1.0*
- *STIX 1.1*
- *STIX 1.1.1*
- *STIX 1.2*
- *Text*
- **Accept password protected archives:** select this checkbox to specify a global password to open any archives retrieved through the incoming feed.

If the archives are password-protected, enter it in the **Archive password** field. The specified password acts as a master password, and it is used to try to unlock and access any archives retrieved with the feed.

Supported archive formats: *.rar*, *.tar*, *.tar.bz2*, *.tar.gz*, *.tar.bz2*, *.tar.z*, *.zip*

Under **Transport configuration**, configure the following settings:

- **Path:** the location/directory/folder on the server or network unit hosting the data source for the feed.
The incoming feed first looks for the specified directory, on a server or a network unit, as the origin to poll data from.
 - If you specify also a *regex pattern*, the path should point to a directory.
 - If you *do not* define a regex pattern, the path should be an absolute pointing to a file, and the file format should match the configured *content type* for the feed.

**Warning:**

You need to *explicitly whitelist mount point paths* to make them accessible to incoming feeds. If you do not whitelist the mount point path an incoming feed should access to retrieve data from, the feed will not be able to return any content.

To whitelist a mount point path open the platform settings file:

```
$ sudo vi /opt/eclecticiq/etc/eclecticiq/platform_settings.py
```

Look for the `MOUNT_POINT_POLL_ALLOWED_DIRECTORIES` parameter.

It is a list that takes valid directory paths as list elements.

Each path in the list points to a location an incoming feed can access as a data source.

Incoming feeds can access files and directories *inside* the specified location, based on the configured access rights of the available assets and resources.

Add as many paths to the list as necessary, then save the file and exit.

Example:

```
# Whitelist specific dirs; specific file types; everything inside a dir subdirs
MOUNT_POINT_POLL_ALLOWED_DIRECTORIES = [ "/mnt/", "/media/", "/media/data/" ]
```

Restart the *platform-api* service to make the changes effective:

```
$ sudo supervisorctl restart platform-api
```

- **Regex pattern:** you can define a regex to include in the incoming feed any file names matching the specified regex pattern.

If you include a regex pattern, the incoming feed looks in the directory specified under **Path** for any files whose name matches the regex pattern, and whose format matches the configured *content type* for the feed.

Your input needs to be a **valid regex pattern** (<https://remram44.github.io/regex-cheatsheet/regex.html>).

Examples of valid regex patterns:

- `+.json`
- `+.pdf`
- `.*`
- `[^/]all_files_with_this_name_but_different_extensions[^\].[a-z0-9]*`

Matching files are added to the feed, so that the platform can ingest their content.

If you do not enter any regex pattern, the feed fetches the base path response body, and it tries to open the base path.

Configure SFTP download transport and content

Set up and configure transport and content types for SFTP download incoming feeds to securely transfer data from selected SFTP servers.

To configure the general options for the SFTP download incoming feed, see [Configure incoming feeds](#).

About SFTP download

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
SFTP download	Structured and unstructured data in JSON, PDF, STIX, and plain text format data found on the designated source location.	The platform ingests retrieved data based on the configured content type for the feed.

Configure transport and content types

Transport type	Allowed content types
SFTP download	CAPEC XML
	EclecticlQ JSON
	EclecticlQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

- From the **Transport type** drop-down menu, select **SFTP download**.

- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.

The selected content type for the feed should match the data source format.

This can vary, depending on the intel sources you retrieve the data from.

The **SFTP download** transport type enables fetching data in the following formats:

- CAPEC XML
- EclecticIQ JSON
- EclecticIQ JSON (Legacy)
- PDF
- STIX 1.0
- STIX 1.1
- STIX 1.1.1
- STIX 1.2
- Text

Under **Transport configuration**, configure the following settings:

- **SFTP server URL**: the location on the SFTP server where the data source for the feed is made available (incoming feeds) or where the feed content is being published to (outgoing feeds).
It needs to adhere to the following format: `sftp://{sftp_server}:{port}/{path_to_target_directory}`
Example: `sftp://sftp.server.com:22/source-data/for-the-feed`
- **Username**: a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Password**: a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Use SSH key**: select this checkbox to enable logging in through SSH to apply this security layer to the incoming feed.
- **SSH private key**: paste into this field the private SSH key you want to use to access the data source of the SFTP download incoming feed.
Example:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEA3Tz2mr7SZiAMfQyuvBjM9Oi..Z1BjP5CE/Wm/Rr500P
RK+Lh9x5eJPo5CAZ3/ANBE0sTK0ZsDGMak2m1g7..3VHqIxFTz0Ta1d+NAj
wnLe4nOb7/eEJbDPkk05ShhBrJGBKKxb8n104o/. .PdzbFMIyNjJzBM2o5y
5A13wiLitEO7nco2WfyYkQzaxCw0AwzlkVHiIyC..71pSzkv6sv+4IDMbT/
XpCo8L6wTarzrywnQsh+etLD6FtTjYbbrvZ8RQM..Hg2qxraAV++HNBmNW
kbJ+q+rsJxQlaipn2M4lGuQJEfIxELFDyd3XpxP..Un/82NZNXlPmRIopXs
2T91jiLZEUKQw+n73j26adTbteuEaPGSrTzxBLR..ySS00wWomUyILqVeti
+PK+aXKwguI6bxLGZ3of0UH+mGsSl0mkp7kYZCm..OTQtfeRqP8rDSC7DgA
kHc5ajYqh04AZNFaxjRo+M3IGICUaOdKnXd0Fda..QwfoaX4QlRTgLqb7AN
ZTzM9WbmnYoXrx17kZlT3lsCgYEAm757XI3WJVj..WoLj1+v48WyoXZpcai
uv9bT4Cj+1XRS+gdKHK+SH7J3x2CRHVS+WH/SVC..DxuybvebDoT0TkKiCj
BWQaGzCaJqZa+POHK0klvS+9ln0/6k539p95tfX..X4TCzbVG6+gJiX0ysZ
Yfehn5MCgYEAkMiKuWHCsVyCab3RUF6XA9gd3qY..fCTIGtSltr5PgFIV+G
engiVoWc/hkj8SBHZZ1n1xLN7Kdf8ySU06MDggB..hJ+gXJKy+gf3mF5Kmj
DtKpjGHQzPF6vOe907y5NQLvVFGXUq/FIJZxB8k..fJdHEm2M4=
-----END RSA PRIVATE KEY-----
```

- **SSH key password**: if your SSH key is password-protected, enter here the password to unlock the SSH key.
If your SSH key is not password-protected, you can leave this field empty.

- **Host authentication mode:** select this checkbox to automatically add and save the new host name and the new host key to the local Paramiko `HostKeys` dictionary.

Configure TAXII inbox transport and content

Set up and configure transport and content types for TAXII inbox incoming feeds to retrieve and process information from specific data sources supporting the TAXII transport type.

To configure the general options for the TAXII inbox incoming feed, see [Configure incoming feeds](#).

About TAXII inbox



Assign unique names to TAXII feeds: TAXII inbox and TAXII poll incoming and outgoing feeds should all have unique names within the platform.

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
TAXII inbox	Structured STIX blobs and packages.	Structured, STIX-compliant entities.

Configure transport and content types

Transport type	Allowed content types
TAXII inbox	CAPEC XML
	EclecticlQ JSON
	EclecticlQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

- From the **Transport type** drop-down menu, select **TAXII inbox**.

- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.
The selected content type for the feed should match the data source format.
This can vary, depending on the intel sources you retrieve the data from.
The **TAXII inbox** transport type enables fetching data in the following formats:

- *CAPEC XML*
- *EclecticIQ JSON*
- *EclecticIQ JSON (Legacy)*
- *PDF*
- *STIX 1.0*
- *STIX 1.1*
- *STIX 1.1.1*
- *STIX 1.2*
- *Text*
- **Accept password protected archives:** select this checkbox to specify a global password to open any archives retrieved through the incoming feed.

If the archives are password-protected, enter it in the **Archive password** field. The specified password acts as a master password, and it is used to try to unlock and access any archives retrieved with the feed.

Supported archive formats: *.rar*, *.tar*, *.tar.bz2*, *.tar.gz*, *.tar.bz2*, *.tar.z*, *.zip*

Under **Transport configuration**, configure the following settings:

- **Public:** select this checkbox to make the incoming feed publicly accessible to all platform users.
By default, this option is deselected.
- **Authorized groups:** from the drop-down menu select one or more platform user groups to authorize them and their members to access to incoming feed.
This option is available when **Public** is deselected.



To set up a TAXII inbox incoming feed, the platform needs a configured and working TAXII inbox service, as well as configured user groups.

Push data to a TAXII inbox

An easy way to push content to a TAXII inbox that acts as a data source for an incoming feed is by feeding it data through a TAXII inbox outgoing feed.

To link the TAXII inbox outgoing feed data source to the TAXII inbox incoming feed data recipient, make sure you correctly set the following options on the outgoing feed configuration page:

- Under **Transport and content > Transport type**, set the type to **TAXII inbox**.
- Under **Transport and content > Content type**, set the same content type the corresponding receiving TAXII inbox incoming feed is expecting to acquire.

- Under **Transport and content > Transport configuration > Destination collection name** , enter the exact name of the corresponding receiving TAXII inbox incoming feed, as specified in the **Feed name** field.

The highlighted options need to match on the feed configuration pages:

Data Configuration

Incoming feeds

Outgoing feeds

Taxonomies

Enrichers

Rules

Transport and content

Transport type *

TAXII inbox

Content type *

EclecticIQ JSON

Feed content *

Datasets *

Please select one or more options

Update strategy *

Please select one

Transport configuration *

Auto Discovery

Inbox service URL *

Destination collection name *

Feed

TAXII version *

1.1

EclecticIQ authentication URL

Username

Password

☐ SSL certificate authentication

☒ Verify SSL

TAXII inbox outgoing feed. This is the data source pushing content to the TAXII inbox incoming feed.

Create incoming feed

General ⓘ

Feed name *

Feed

Organization *

Source reliability ⓘ

Please select one ▼

☐ Require valid signature☐ Skip extraction of observables from unstructured text

Transport and content ⓘ

Transport type * ⓘ

TAXII inbox × ▼

Content type *

EclecticIQ JSON × ▼

☐ Accept password protected archives

Transport configuration *

☐ Public

Authorized groups *

Please select one or more options ▼

TAXII inbox incoming feed. This is the data recipient obtaining content from the TAXII inbox outgoing feed.

Configure TAXII poll transport and content

Set up and configure transport and content types for TAXII poll incoming feeds to poll discovery services to retrieve and process information from specific data sources supporting the TAXII transport type.

To configure the general options for the TAXII poll incoming feed, see [Configure incoming feeds](#).

About TAXII poll



Assign unique names to TAXII feeds: TAXII inbox and TAXII poll incoming and outgoing feeds should all have unique names within the platform.

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
TAXII poll	Structured STIX blobs and packages.	Structured, STIX-compliant entities.

Configure transport and content types

Transport type	Allowed content types
TAXII poll	CAPEC XML
	EclecticIQ JSON
	EclecticIQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

- From the **Transport type** drop-down menu, select **TAXII poll**.

- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.
The selected content type for the feed should match the data source format.
This can vary, depending on the intel sources you retrieve the data from.
The **TAXII poll** transport type enables fetching data in the following formats:

- *CAPEC XML*
- *EclecticIQ JSON*
- *EclecticIQ JSON (Legacy)*
- *PDF*
- *STIX 1.0*
- *STIX 1.1*
- *STIX 1.1.1*
- *STIX 1.2*
- *Text*

- **Accept password protected archives:** select this checkbox to specify a global password to open any archives retrieved through the incoming feed.

If the archives are password-protected, enter it in the **Archive password** field. The specified password acts as a master password, and it is used to try to unlock and access any archives retrieved with the feed.

Supported archive formats: *.rar, .tar, .tar.bz2, .tar.gz, .tar.bz2, .tar.z, .zip*

Under **Transport configuration**, configure the following settings:

- **Auto Discovery:** enter the URL of a TAXII discovery service the feed can send a request to in order to determine the available services and poll for updates.
Example: *http://hailataxii.com/taxii-discovery-service*
- **Polling service URL:** enter the URL of a TAXII poll service the feed can poll for updates.
Example: *http://hailataxii.com/taxii-polling-service*
- **Collection name:** enter the name of the data collection you want to poll and use as a source for the feed.
The collection you define here needs to be available through the auto-discovery or the polling services.
Example: *guest.Abuse_ch*
- **TAXII version:** specify the TAXII version the data source service uses to publish data.
Example: *1.1*
- **Extra headers:** the TAXII server hosting the data source for the feed may require passing additional HTTP headers in the request.
Click **+ Add** or **+ More** to include in the request one or more additional HTTP headers, along with the corresponding values.
- **Subscription ID:** enter the name, label or ID identifying the subscription session that makes the incoming feed content available.
Usually, you need to sign up to a service to initiate such a session.
- **Ingest messages starting from:** select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.
- **Days per poll:** if you select a start date to poll data from, you can enter an integer to specify the maximum number of days to poll at a time.
This enables polling in multiple smaller batches, instead of a single batch, starting from the selected initial date.
This option works only if you select an ingestion start date.

- **Verify SSL:** if the data source TAXII server requires an SSL certificate to authenticate and to access its TAXII services, you can select this checkbox to test the SSL connection and to verify that it works correctly.
 - **SSL CA bundle file path:** enter the path to the CA bundle file containing the root, intermediate, and public certificates for the SSL authentication.
- **Basic authentication:** if the data source TAXII server requires basic authentication to access the corresponding TAXII services, select this checkbox to fill out the required information.
 - **Username:** a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
 - **Password:** a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
 - **EclecticIQ authentication URL:** the EclecticIQ Platform URL including the endpoint that takes the user name and password inputs to send them to the authentication mechanism.
Example: *http://\${platform_host}/api/auth*

- **SSL certificate authentication:** if the TAXII server requires an SSL certificate to authenticate and to authorize access to the corresponding TAXII services, select this checkbox to fill out the required information.
- **SSL certificate:** copy-paste the content of a valid SSL certificate to authenticate.
SSL certificate file format: PEM
Example:

```
-----BEGIN CERTIFICATE REQUEST-----
MIICvDCCAAQCAQAwdzELMAkGA1UEBhMCVVMxDTALBgNVBAgMBFV0YWgxDzANBgNV
BACMBkxpbmRvbWJEWMBQGA1UECgwNRGlnaUNlcnQgSW5jLjERMA8GA1UECwwIRGln
aUNlcnQxHTAbBgNVBAMMFjY4YUw1bGUuZGlnaWNlcnQuY29tMIIBIjANBgkqhkiG
9w0BAQEFAAOCAQ8AMIIBCgKCAQEAA8+To7d+2kPWeBv/orU3LVbJwDrSQbeKamCmo
wp5bqDxIwV20zqRb7APUOKY0VEFFOEQs6T6gImnIo1hbiH6m4zgZ/CPvWBokZc+c
1Po2EmvBz+AD5sBdT5kzGQA6NbWYzGldxRthNLOslefOhdnWFuhI162qmcflgpiI
Wduwq4C9f+YkeJhNn9dF5+owm8cOQmDrV8NNdiTqin8q3qYAHHJRW28glJUCZkTZ
wIaSR6crBQ8TbYNE0dc+Caa3DOIkz1EOsHWzTx+n0zKfqcBgXi4DJx+C1bjptYPR
BPZL8DAeWuA8ebudVT44yEp82G96/Ggcf7F33xMxe0yc+Xa6owIDAQABoAAwDQYJ
KoZlHvcNAQEFAADggEBAB0kcrFccSmFDmxx0Ne01UIqSsDqHgL+XmHTXJwre6D
hJSZwbvEtOKG3+dr4Fs11WuUNT5qcLsx5a8uk4G6AKHMzuhLsJ7XZjgmQXGECpY
Q4mC3yT3Z0CGpIXbw+iP3lmeEXgaQL0Tx5LF1/okKbKYWiQNiYKWOMj7ZR/wxWg/
ZDGRs55xuoeLDJ/ZRff9bI+IaCud1YrfYcHl3G87Av+r49YVwqRDT0VDV7uLgqn
29XI1PpVUNCPQgn9p/eX6Qo7vpDaPybRtA2R7XLKjQaF9oXWeCUqylhvJac9QF02
970blalPpOZ7mWiEuJwJBpIi6a9M9G30nUo391Bilw=
-----END CERTIFICATE REQUEST-----
```

- **SSL key:** copy-paste the content of a valid SSL key to authenticate.
SSL key file format: PEM
Example:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAA3Tz2mr7SZiAMfQyuvBjM9Oi..Z1BjP5CE/Wm/Rr500P
RK+Lh9x5eJPo5CAZ3/ANBE0sTK0ZsDGMak2m1g7..3VHqIXFTz0Ta1d+NAj
wnLe4n0b7/eEJbDPkk05ShhBrJGBKKxb8n104o/..PdzbFMIYnJzBM2o5y
5A13wiLitEO7nco2WfyYkQzaxCw0AwzlkVHiIyC..71pSzkv6sv+4IDMbT/
XpCo8L6wTarzrywnQsh+etLD6FtTjYbbrvZ8RQM..Hg2qxraAV++HNBmNW
kbJ+q+rsJxQlaipn2M4lGuQJEfIxELFDyd3XpxP..Un/82NZNX1PmRIopXs
2T91jiLZEUKQw+n73j26adTbteuEaPGSrTZxBLR..yss00wWomUyILqVeti
+PK+aXKwguI6bxLGZ3of0UH+mGsS10mkp7kYZCm..OTQtfeRqP8rDSC7DgA
kHc5ajYqh04AZNFaxjRo+M3IGICUaOdKnXd0Fda..QwfoaX4QLRTgLqb7AN
ZTzM9WbmnYoXrx17kZlT3lsCgYEAm757XI3WJVj..WoLj1+v48WyoXZpcai
uv9bT4Cj+1XRS+gdKHK+SH7J3x2CRHVS+WH/SVC..DxuybvebDoT0TkKiCj
BWQaGzCaJqZa+POHK0klvS+9ln0/6k539p95tfX..X4TCzbVG6+gJiX0ysZ
Yfehn5MCgYEAkMiKuWHCsVyCab3RUF6XA9gd3qY..fCTIGtS1tR5PgFIV+G
engiVoWc/hk78SBHzzln1LN7KdF8ySU06MDggB..hJ+gXJKy+gf3mF5KmJ
DtKpjGHQzPF6v0E907y5NQLvVFGXuq/FIJZxB8k..fJdHEm2M4=
-----END RSA PRIVATE KEY-----
```

- **SSL key password:** enter the SSL password or passphrase for the SSL key. This field is masked.

Configure Amazon S3 poll transport and content

Set up and configure transport and content types for Amazon S3 poll incoming feeds to securely transfer data from selected Amazon S3 buckets.

To configure the general options for the Amazon S3 poll incoming feed, see [Configure incoming feeds](#).

About Amazon S3 poll

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Amazon S3 poll	All data found on the designated source S3 bucket.	The platform ingests retrieved data based on the configured content type for the feed. After successfully retrieving and ingesting the data, it is removed from the source bucket.

Configure transport and content types

Transport type	Allowed content types
Amazon S3 poll	CAPEC XML
	EclecticlQ JSON
	EclecticlQ JSON (Legacy)
	Email message
	PDF
	STIX 1.0
	STIX 1.1
	STIX 1.1.1
	STIX 1.2
	Text

- From the **Transport type** drop-down menu, select **Amazon S3 poll**.

- From the **Content type** drop-down menu, select the appropriate content type for the data you want to ingest through the incoming feed.
The selected content type for the feed should match the data source format you want to retrieve with the feed.
The **Amazon S3 poll** transport type enables fetching data in the following formats:
 - *CAPEC XML*
 - *Eclectic JSON*
 - *Eclectic JSON (Legacy)*
 - *Email message*
 - *PDF*
 - *STIX 1.0*
 - *STIX 1.1*
 - *STIX 1.1.1*
 - *STIX 1.2*
 - *Text*
- **Secret key**: sign up to Amazon Web Services, and then create one or more accounts, as necessary, to use their S3 data storage service. The secret key is part of your authentication credentials to log in to and access Amazon S3 services.
- **Access key**: along with your secret key, the access key enables you to authenticate to access Amazon S3 services.
- **Bucket**: the name of the Amazon S3 bucket to use as a data source for the feed.
Buckets (<https://docs.aws.amazon.com/amazons3/latest/dev/usingbucket.html>) are data containers in the S3 environment.
Buckets are region-specific, and their names need to comply with standard DNS naming conventions.
The default format of the URL to access a bucket is *https://{bucket_name}.s3-{aws-region}.amazonaws.com*
- **Path**: the path to the target directory where the source data for the incoming feed is stored, relative to the bucket root.
Example: */intel/actors/hacktivists*

Configure Anubis Cyberfeed transport and content

Set up and configure transport and content types for AnubisNetworks Cyberfeed incoming feeds to retrieve and process information on compromised machines and DNS servers, as well as compromised web sites and malware files.

To configure the general options for the Anubis Cyberfeed incoming feed, see [Configure incoming feeds](#).

About Anubis Cyberfeed

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Infection detection Bank Trojans	Metadata from communication involving Trojan-infected machines.	An ID reference to the TTP containing information on the identified Trojan family.
		An indicator related with ID reference to the TTP, containing the Command and Control system address as server name or server address, as well as the first and last time the threat was sighted.
		A sighting related to the indicator, containing the IP address or the domain name of the compromised machine, HTTP request details like request method, cookies, HTTP user agent, the client IP address passed on to the server via a XFF HTTP header, any additional arguments, as well as the first and last time the threat was sighted.
Infection detection DNS malware	Metadata from communication involving Trojan-infected DNS servers.	An ID reference to the TTP containing information on the identified Trojan family.
		An indicator related with ID reference to the TTP, containing the Command and Control system address as server name or server address.
		A sighting related to the indicator, containing the IP address of the compromised machine, and the DNS query type (https://en.wikipedia.org/wiki/list_of_dns_record_types).
Compromised systems website analysis	Indicators of compromise concerning malware-infected web sites.	An indicator with the malware-targeted URL, a description, any available behavior signatures, as well as the first and last time the threat was sighted.
		Observables for any found IP addresses, domain names, or hashes.

Feed	Ingested data	Processed data
Compromised systems malware analysis	Indicators of compromise concerning analyzed malware samples.	An indicator with details about the malware file, a description, any available behavior signatures, as well as the first and last time the threat was sighted.
		Observables for any found file sizes, file types, or file hashes.

By default, the entities the platform creates after processing the ingested data from Anubis Cyberfeed incoming feeds include the following properties:

Property	Default value	Entities
Confidence	High	TTPs, indicators, sightings
Likely impact	High	Indicators
Impact	High	Sightings

Configure transport and content types

Transport type	Allowed content types
Anubis Cyberfeed	Anubis Cyberfeed JSON

- From the **Transport type** drop-down menu, select **Anubis Cyberfeed**.
- From the **Content type** drop-down menu, select **Anubis Cyberfeed JSON**.
The **Anubis Cyberfeed** transport type supports only the **Anubis Cyberfeed JSON** content type.

The organization providing the source data for the incoming feed is AnubisNetworks.

Under **Transport configuration**, configure the following settings:

- **API URL**: the URL pointing to the API endpoint exposing the service that makes the data available for retrieval through the feed. Contact the intel provider of the incoming feed to obtain this information.
- **API key**: contact AnubisNetworks to receive an API key, and then enter it in the corresponding input field.
- **Infection detection bank trojans**: select this checkbox to fetch metadata from data flows between machines compromised with Trojans and the AnubisNetworks sinkhole platform.
This channel provides information such as IP addresses, Trojan family, request metadata, request payload, and pattern verification.
- **Infection detection DNS malware**: select this checkbox to fetch metadata from DNS servers on Trojans trying to contact the AnubisNetworks sinkhole platform. This information allows to detect potential compromises when Command and Control communication is intercepted before it reaches the recipient Command and Control center.
This channel provides information such as DNS request details, origin IP addresses, and Trojan family.
- **Compromised systems website analysis**: select this checkbox to fetch metadata to detect compromised web sites and servers on the specified networks, to detect and profile compromised hosts, and to support mitigation by providing indicators and observables.

- **Compromised systems malware analysis:** select this checkbox to fetch metadata to detect malicious file infections on the specified networks, and to support mitigation by providing indicators and observables.

Configure BFK transport and content

Set up and configure transport and content types for BFK incoming feeds to retrieve and process reports on cyber threats and activities, as well as information on NIDs (Network Intrusion Detections).

To configure the general options for the BFK API incoming feed, see [Configure incoming feeds](#).

About BFK API

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
BFK API	Reports and NIDs (Network Intrusion Detections).	Ingested reports are saved as report entities in the platform, whereas ingested NIDs produce indicators with linked TTPs.

Configure transport and content types

Transport type	Allowed content types
BFK API	BFK Threat Intelligence JSON

- From the **Transport type** drop-down menu, select **BFK API**.
- From the **Content type** drop-down menu, select **BFK Threat Intelligence JSON**.
The **BFK API** transport type supports only the **BFK Threat Intelligence JSON** content type.

The organization providing the source data for the incoming feed is BFK edv-consulting.

Under **Transport configuration**, configure the following settings:

- **Username**: a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **Password**: a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
Contact the intelligence provider to subscribe to the service and to obtain the required authentication and authorization credentials.
- **Polling start time**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days.

Configure Cisco Threat Grid Curated Feed transport and content

Set up and configure transport and content types for Cisco Threat Grid Curated Feed incoming feeds to retrieve and process information on Trojan-infected machines and servers, on malware-infected DNS servers and web sites, on parked domains, DNS sinkholes, and stolen certificates.

To configure the general options for the Cisco Threat Grid Curated Feed incoming feed, see [Configure incoming feeds](#).

About Cisco Threat Grid Curated Feed

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data
banking-dns	Banking Trojan network communications: (meta)data from communication involving Trojan-infected machines.
dll-hijacking-dns	Data obtained from the analysis of samples leveraging DLL sideloading and/or hijacking techniques
doc-net-com-dns	Document (PDF, Office) network communications: (meta)data from communication involving infected document files.
downloaded-pe-dns	Data obtained from the analysis of samples downloading executables over the network.
dynamic-dns	Data obtained from the analysis of samples leveraging dynamic DNS providers.
irc-dns	Data obtained from Internet Relay Chat (IRC) network communications.
modified-hosts-dns	Information about modified Windows hosts files.
parked-dns	Information about parked domains resolving to RFC1918 (https://tools.ietf.org/html/rfc1918), localhost and broadcast addresses.
public-ip-check-dns	Check For Public IP Address Network Communications.
ransomware-dns	Data obtained from the analysis of samples communicating with ransomware servers.
rat-dns	Information about remote access Trojans (RAT), and any communications with their Command and Control systems.
sinkholed-ip-dns	DNS entries obtained from the analysis of samples communicating with known DNS sinkholes.
stolen-cert-dns	DNS entries observed in samples signed with a stolen certificate.

The ingested data produce indicators with embedded observables, where each observable represents an indicator of compromise (IOC).

Configure transport and content types

Transport type	Allowed content types
Cisco Threat Grid Curated Feed	STIX 1.2

- From the **Transport type** drop-down menu, select **Cisco Threat Grid Curated Feed**.
- From the **Content type** drop-down menu, select **STIX 1.2**.
Currently, the **Cisco Threat Grid Curated Feed** transport type supports only the **STIX 1.2** content type.
- **API URL**: the URL pointing to the API endpoint exposing the service that makes the data available for retrieval through the feed. Contact the intel provider of the incoming feed to obtain this information.
This field should be pre-populated with <https://panacea.threatgrid.com/api/v3/>.
- **API key**: contact Cisco to receive an API key, and then enter it in the corresponding input field.
- **Feed type**: from the drop-down menu select the data source you want the incoming feed to retrieve data from. The available channels are:
 - **Banking Trojan Network Communications**
 - **Feed contains Domains communicated to by samples leveraging DLL Sideloads and/or hijacking techniques**
 - **Document (PDF, Office) Network Communications**
 - **Samples Downloading Executables Network Communications**
 - **Samples Leveraging Dynamic DNS Providers**
 - **Internet Relay Chat (IRC) Network Communications**
 - **Modified Windows Hosts File Network Communications**
 - **Parked Domains resolving to RFC1918, Localhost and Broadcast Addresses**
 - **Check For Public IP Address Network Communications**
 - **Samples Communicating with Ransomware Servers**
 - **Remote Access Trojan (RAT) Network Communications**
 - **DNS entries for samples communicating with a known dns sinkhole**
 - **DNS Entries observed from samples signed with a stolen certificate**
- **Ingest feed starting from**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.
If you do not specify any start date, the default start date is 6 months in the past. This means that if you leave this field empty, the incoming feed will fetch data as old as 6 months until the present time.

Configure Cisco Threat Grid Samples API transport and content

Set up and configure transport and content types for Cisco Threat Grid Samples API incoming feeds to retrieve and process information on compromised IP addresses, domains, hashes, registry keys, and network streams.

To configure the general options for the Cisco Threat Grid Samples API incoming feed, see [Configure incoming feeds](#).

About Cisco Threat Grid Samples API

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Samples API	Data obtained from the analysis of submitted samples. Samples can be files, file types, running processes, URLs, IP domains, hashes, and registry keys.	Observables, where each observable represents an indicator of compromise (IOC).

Configure transport and content types

Transport type	Allowed content types
Cisco Threat Grid Samples API	Cisco Threat Grid JSON

- From the **Transport type** and **Content type** drop-down lists, choose the appropriate transport and content types for the data, based on the available options.
The selected content type for the feed should match the data source format.
This can vary, depending on the intel source(s) you retrieve the data from.

The intel provider for the feed is Cisco Threat Grid Samples API.

Under **Transport configuration**, configure the following settings:

- **API URL**: the URL pointing to the API endpoint exposing the service that makes the data available for retrieval through the feed. Contact the intel provider of the incoming feed to obtain this information.
- **API key**: contact Cisco to receive an API key, and then enter it in the corresponding input field.
- **Confidence threshold**: integer. Defines a minimum value to filter out and exclude from the feed any threats whose reliability level, that is, a score representing the trustworthiness of the information, is lower than the specified value.
 - Allowed range: 0-100
 - Default value: 80

- **Severity threshold:** integer. Defines a minimum value to filter out and exclude from the feed any threats whose severity level, that is, a score representing the potential risk they may pose, is lower than the specified value.
 - Allowed range: *0-100*
 - Default value: *80*
- **Threat score threshold:** integer. Defines a minimum value to filter out and exclude from the feed any threats ranking lower than the specified value. It represents a threat potential aggressiveness and maliciousness.
 - Allowed range: *0-100*
 - Default value: *80*

Setting thresholds enables conditional execution. For example, if an entity in the feed holds a value exceeding the predefined threshold, it can trigger a process to execute a follow-up action.

- **Ingest samples starting from:** select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.
If you do not specify any start date, the default start date is 6 months in the past. This means that if you leave this field empty, the incoming feed will fetch data as old as 6 months until the present time.

By default, the following checkboxes are preselected to include these observables in the feed:

- **Organization only:** select this checkbox to enable the enricher to check and display only submitted samples created by the organization the current user belongs to. That is, the organization needs to be the author of the submitted samples. When selected, this field is validated against the API key value granting access to the service. This checkbox is not preselected by default.
- **Fetch IPs:** includes samples about compromised IP addresses.
- **Fetch URLs:** includes samples about compromised URLs.
- **Fetch domains:** includes samples about compromised domain names.
- **Fetch artifacts:** includes samples about compromised artifacts; for example, file hash values.
- **Fetch registry keys:** includes samples about compromised (Windows) registry key data.
- **Fetch network streams:** includes samples about network stream information exchanged among compromised machines.

Configure Crowdstrike Falcon Intelligence Indicator transport and content

Set up and configure transport and content types for Crowdstrike Falcon Intelligence Indicator incoming feeds to retrieve and process information on indicators, such as compromised devices, malicious domains, hashes, and more.

To configure the general options for the Crowdstrike Falcon Intelligence Indicator incoming feed, see [Configure incoming feeds](#).

About Crowdstrike Falcon Intelligence Indicator

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Crowdstrike Falcon Intelligence Indicator	Indicators found in their environment, related to entities such as threat actors or other indicators.	Indicators retrieved from the Falcon Intelligence platform such as compromised devices, malicious domains, hashes, and so on starting from the specified polling date.

Configure transport and content types

Transport type	Allowed content types
Crowdstrike Falcon Intelligence Indicator Feed	Crowdstrike Indicator JSON

- From the **Transport type** drop-down menu, select **Crowdstrike Falcon Intelligence Indicator Feed**.
- From the **Content type** drop-down menu, select **Crowdstrike Indicator JSON**.
The **Crowdstrike Falcon Intelligence Indicator** transport type supports only the **Crowdstrike Indicator JSON** content type.

The organization providing the source data for the incoming feed is Crowdstrike.

Under **Transport configuration**, configure the following settings:

- **API ID**: contact Crowdstrike to receive an API ID, and then enter it in the corresponding input field.
You need a valid API ID and a corresponding API key as authentication credentials to access the Crowdstrike Falcon Intel API and to consume it.
- **API key**: contact Crowdstrike to receive an API key, and then enter it in the corresponding input field.
- **Polling start time**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days.

You need a valid subscription to CrowdStrike Falcon Premium Intel to access CrowdStrike intelligence publications.

©2018 by EclecticIQ BV. All rights reserved.
Last generated on Mar 6, 2018

Configure Crowdstrike Falcon Intelligence Reports transport and content

Set up and configure transport and content types for Crowdstrike Falcon Intelligence Report incoming feeds to retrieve and process Crowdstrike intelligence reports, periodic reports, and tippers.

To configure the general options for the Crowdstrike Falcon Intelligence Reports incoming feed, see [Configure incoming feeds](#).

About Crowdstrike Falcon Intelligence Reports

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Crowdstrike Falcon Intelligence Reports	Crowdstrike Intelligence Reports (CSIR), Periodic Reports (CSMR), Tippers (CSIT)	Reports retrieved from the Falcon Intelligence platform in JSON format and as PDF attachments.

Configure transport and content types

Transport type	Allowed content types
Crowdstrike Falcon Intelligence Reports Feed	Crowdstrike report JSON

- From the **Transport type** drop-down menu, select **Crowdstrike Falcon Intelligence Reports Feed**.
- From the **Content type** drop-down menu, select **Crowdstrike report JSON**.
The **Crowdstrike Falcon Intelligence Reports** transport type supports only the **Crowdstrike report JSON** content type.

The organization providing the source data for the incoming feed is Crowdstrike.

Under **Transport configuration**, configure the following settings:

- **API ID**: contact Crowdstrike to receive an API ID, and then enter it in the corresponding input field.
You need a valid API ID and a corresponding API key as authentication credentials to access the Crowdstrike Falcon Intel API and to consume it.
- **API key**: contact Crowdstrike to receive an API key, and then enter it in the corresponding input field.
- **Polling start time**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days. - Select one or more checkboxes to include one or more report types to ingest through the incoming feed: - **Include Intelligence reports**: reports about specific topics, cases, and scenarios. - **Include Tippers**: tips and tricks to improve cyber security defenses.

You need a valid subscription to CrowdStrike Falcon Premium Intel to access CrowdStrike intelligence publications.

©2018 by EclecticIQ BV. All rights reserved.
Last generated on Mar 6, 2018

Configure Crowdstrike Falcon Intelligence Threat Actor transport and content

Set up and configure transport and content types for Crowdstrike Falcon Intelligence Threat Actor incoming feeds to retrieve and process intelligence on threat actors, their TTPs, and any campaigns they may be related to.

To configure the general options for the Crowdstrike Falcon Intelligence Threat Actor incoming feed, see [Configure incoming feeds](#).

About Crowdstrike Falcon Intelligence Threat Actor

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Crowdstrike Falcon Intelligence Threat Actor Feed	Queries the Crowdstrike API to retrieve information on threat actors based on actor names, target countries, target industries, and (short) descriptions.	Threat actor entities, related TTPs, indicators, and campaigns, as well as related observables to represent actor ID, target country, target industry, and targeted victim(s).

Configure transport and content types

Transport type	Allowed content types
Crowdstrike Falcon Intelligence Threat Actor Feed	Crowdstrike actor JSON

- From the **Transport type** drop-down menu, select **Crowdstrike Falcon Intelligence Threat Actor Feed**.
- From the **Content type** drop-down menu, select **Crowdstrike actor JSON**.
The **Crowdstrike Falcon Intelligence Threat Actor Feed** transport type supports only the **Crowdstrike actor JSON** content type.

The organization providing the source data for the incoming feed is Crowdstrike.

Under **Transport configuration**, configure the following settings:

- **API ID**: contact Crowdstrike to receive an API ID, and then enter it in the corresponding input field.
You need a valid API ID and a corresponding API key as authentication credentials to access the Crowdstrike Falcon Intel API and to consume it.
- **API key**: contact Crowdstrike to receive an API key, and then enter it in the corresponding input field.
- **Polling start time**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days.

You need a valid subscription to CrowdStrike Falcon Premium Intel to access CrowdStrike intelligence publications.

©2018 by EclecticIQ BV. All rights reserved.
Last generated on Mar 6, 2018

Configure CVE Search API transport and content

Set up and configure transport and content types for CVE Search API incoming feeds to retrieve and process information on common software and hardware vulnerabilities, along with the corresponding exposures.

To configure the general options for the CVE Search API incoming feed, see [Configure incoming feeds](#).

About CVE Search API

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
CVE Search API	CVE information about common software and hardware vulnerabilities, and exposures.	Exploit target entities. The entity ID is derived from the CVE ID.

Configure transport and content types

Transport type	Allowed content types
CVE Search API	CVE Search JSON

- From the **Transport type** drop-down menu, select **CVE Search API**.
- From the **Content type** drop-down menu, select **CVE Search JSON**.
The **CVE Search API** transport type supports only the **CVE report JSON** content type.

The organization providing the source data for the incoming feed is the Computer Incident Response Center Luxembourg (CIRCL).

Under **Transport configuration**, configure the following settings:

- **CVE API URL**: the URL pointing to the API endpoint exposing the service that grants access to the enricher data.
Contact the intel provider to subscribe to the service and to obtain this information.
For this incoming feed it corresponds to <https://cve.circl.lu/api/last>.

By default, every time you run the feed, it ingests the most recent 30 CVE entries.
Run this feed on a regular basis to keep up to date with the latest CVE definitions.

Configure FireEye iSIGHT Intelligence Report API transport and content

Set up and configure transport and content types for FireEye iSIGHT Intelligence Report API incoming feeds to retrieve and process STIX reports on vulnerabilities, malware, and other threats such as threat actors, strategies, tactics, and techniques.

To configure the general options for the FireEye iSIGHT Intelligence Report API incoming feed, see [Configure incoming feeds](#).

About FireEye iSIGHT Intelligence Report API

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
FireEye iSIGHT Intelligence Report API	FireEye iSIGHT intelligence reports published since a specific start date.	STIX reports on vulnerabilities, malware, and threats such as threat actors, strategies, tactics, and techniques.

Configure transport and content types

Transport type	Allowed content types
FireEye iSIGHT Intelligence Report API	STIX 1.1.1

- From the **Transport type** drop-down menu, select **FireEye iSIGHT Intelligence Report API**.
- From the **Content type** drop-down menu, select **STIX 1.1.1**.

The intel provider for the feed is iSIGHT API.

Under **Transport configuration**, configure the following settings:

- **FireEye API URL**: the URL pointing to the API endpoint exposing the service that grants access to the incoming feed data. Contact the intel provider to subscribe to the service and to obtain this information.
- **API public key**: enter the API public key to authorize your account. To access the iSIGHT API you need to authenticate using a public and a private key. Make sure both keys are securely stored.
- **API private key**: enter the API private key to authorize your account. To access the iSIGHT API you need to authenticate using a public and a private key. Make sure both keys are securely stored.
- **Ingest feed starting from**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 89 days.

If you set an ingestion start date at a point in time further back in the past, the feed sends multiple requests to retrieve the data. - **Include Malware intelligence type**: select this checkbox to include in the feed intelligence reports about malware.

- **Include Threat intelligence type**: select this checkbox to include in the feed intelligence reports about threats such as threat actors, strategies, tactics, techniques, campaigns, and so on. - **Include Vulnerability intelligence type**: select this checkbox to include in the feed intelligence reports about vulnerabilities.

Configure Group-IB transport and content

Set up and configure transport and content types for Group-IB incoming feeds to retrieve and process information on compromised logins and (email) accounts, payment cards, mobile devices, money mules, and so on.

To configure the general options for the Group-IB incoming feed, see [Configure incoming feeds](#).

About Group-IB

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data
accs	Information on compromised login and password credentials.
cards	Information on compromised payment bank cards.
imei	Signatures of compromised or infected mobile devices.
mules	Information on fraudulent bank accounts (money mules).
files	Information on compromised or infected files.
phishing	Information on phishing resources.
ddos	Information on on DDoS attacks.
hacktivism	Information on hacktivist groups and hacktivism.
sample	Information on malicious programs and applications.
malware	Information on malware.
cnc	Indicators to identify Command and Control servers.
threats	Description of threats, including indicators of compromise.
indicator_tor_nodes	Indicators to identify Tor nodes.
indicator_open_proxy	Indicators to identify open proxies.
indicator_socks_proxy	Indicators to identify SOCKS proxies.

Configure transport and content types

Transport type	Allowed content types
TAXII poll	STIX 1.1.1

- From the **Transport type** drop-down menu, select **TAXII poll**.
- From the **Content type** drop-down menu, select **STIX 1.1.1**.
- **Accept password protected archives:** select this checkbox to specify a global password to open any archives retrieved through the incoming feed.

If the archives are password-protected, enter it in the **Archive password** field. The specified password acts as a master password, and it is used to try to unlock and access any archives retrieved with the feed.

Supported archive formats: *.rar, .tar, .tar.bz2, .tar.gz, .tar.bz2, .tar.z, .zip*

Under **Transport configuration**, configure the following settings:

- **Auto Discovery:** if applicable, enter the URL of a Group-IB TAXII discovery service the feed can send a request to in order to determine the available services and poll them for updates.
The current TAXII discovery URL to retrieve the available Group-IB services is *https://bt.group-ib.com/taxii/services/discovery*.
- **Polling service URL:** enter the URL of the Group-IB TAXII polling service the feed can check for updates.
The current TAXII polling URL to retrieve Group-IB content is *https://bt.group-ib.com/taxii/services/poll*.
- **Collection name:** enter the name of the data collection you want to poll and use as a source for the feed.
The collection you specify here needs to be available through the auto-discovery or the polling services.

Group-IB offers a range of thematic intel collections focusing on cyber threats and targets such as money mules, compromised banking accounts and payment cards, Tor nodes, and so on.

Collection availability and access to collections may depend on the terms and conditions of your subscription to Group-IB services.

- **TAXII version:** specify the TAXII version the service uses to publish data.
Example: *1.1*
- **Extra headers:** click **+ Add** or **+ More** to include in the request additional HTTP headers, along with the corresponding values.
Group-IB requires the following extra headers:
 - **X-Auth-Key:** use this header to pass the key or user name you can obtain when subscribing to Group-IB services.
 - **X-Auth-Login:** use this header to pass the password you can obtain when subscribing to Group-IB services.
- **Subscription ID:** enter the name, label or ID identifying the Group-IB subscription session that authorizes accessing the source content available for retrieval through the feed.
- **Ingest messages starting from:** select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.
- **Days per poll:** if you select a start date to poll data from, you can enter an integer to specify the maximum number of days to poll at a time.
This enables polling in multiple smaller batches, instead of a single batch, starting from the selected initial date.
This option works only if you select an ingestion start date.
- **Verify SSL:** if the Group-IB TAXII server requires an SSL certificate to authenticate and to access the corresponding TAXII services, you can select this checkbox to test the SSL connection and to verify that it works correctly.
 - **SSL CA bundle file path:** enter the path to the CA bundle file containing the root, intermediate, and public certificates for the SSL authentication.

- **Basic authentication:** leave the checkbox deselected. You do not need to configure basic authentication for this STIX/TAXII incoming feed setup, since Group-IB requires passing authentication credentials through extra HTTP headers.
- **SSL certificate authentication:** if the TAXII server requires an SSL certificate to authenticate and to authorize access to the corresponding TAXII services, select this checkbox to fill out the required information.
- **SSL certificate:** copy-paste the content of a valid SSL certificate to authenticate.
SSL certificate file format: PEM
Example:

```
-----BEGIN CERTIFICATE REQUEST-----
MIICvDCCAAQCAQAwdzELMAkGA1UEBhMCVVMxDTALBgNVBAgMBFV0YWgxDzANBgNV
BACMBkxpbmRvbWJEWMBQGA1UECgwNRRGlnaUNlcnQgSW5jLjERMA8GA1UECwwIRGln
aUNlcnQxHTAbBgNVBAMMFjY4YUw1bGUuZGlnaWNlcnQuY29tMIIBIjANBgkqhkiG
9w0BAQEFAAOCAQ8AMIIBCgKCAQEA8+To7d+2kPWeBv/orU3LVbJwDrSQbeKamCmo
wp5bqDxIwV20zqRb7APUOKY0VEFFOEs6T6gImnIolhbiH6m4zgZ/CPvWBOkZc+c
1Po2EmvBz+AD5sBdT5kzGQA6NbWyZGldxRthNLOslefOhdnWFuhI162qmcflgpiI
WDuwq4C9f+YkeJhNn9dF5+owm8cOQmDrV8NNdiTqin8q3qYAHHJRW28glJUCZkTZ
wIaSR6crBQ8TbYNE0dc+Caa3DOIkz1EOsHWzTx+n0zKfqcbgXi4DJx+ClbjptYPR
BPZL8DAeWuA8ebudVT44yEp82G96/Ggcf7F33xMxe0yc+Xa6owIDAQABoAAwDQYJ
KoZiHvcNAQEFAADgEBABoKcrFccSmFDmxox0Ne01UIqSsDqHgI+XmHTXJwre6D
hJSZwbvEtOKG3+dr4Fs11WuUNT5qcLsx5a8uk4G6AKHMzuhLsJ7XZjgmQXGECpY
Q4mC3yT3ZocGPiXbw+iP3lmeEXgaQL0Tx5LF1/okKbKYwIQNiYKWOMj7ZR/wxWg/
ZDGRs55xuoeLDJ/ZRFF9bI+IaCUd1YrfYcHl13G87Av+r49YVwqRDT0VDV7uLgqn
29XI1PpVUNCPQgn9p/eX6Qo7vpDaPybRtA2R7XLKjQaF9oXWeCUqylhvJac9QFO2
97Ob1alPpPoZ7mWiEuJwJBPii6a9M9G30nUo391Bilw=
-----END CERTIFICATE REQUEST-----
```

- **SSL key:** copy-paste the content of a valid SSL key to authenticate.
SSL key file format: PEM
Example:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAQAQEA3Tz2mr7SZiAMfQyuvBjM9Oi..Z1BjP5CE/Wm/Rr500P
RK+Lh9x5eJPo5CAZ3/ANBE0sTK0ZsDGMak2mlg7..3VHqIxFTz0Ta1d+NAj
wnLe4nOb7/eEjBDpk05ShhBrJGBKKxb8n104o/.PdzbFMIyNjJzBM2o5y
5A13wiLitEO7nco2WfyYkQzaxCw0AwzlkVHiIyC..71pSzkv6sv+4IDMbT/
XpCo8L6wTarzrywnQsh+etLD6FtTjYbbrvZ8RQM..Hg2qxraAV++HNBmNW
kbJ+q+rsJxQlaipn2M4lGuQJEfIxELFDyd3XpxP..Un/82NZNXlPmRIopXs
2T91jiLzEUKQw+n73j26adTbteuEaPGSrTZxBLR..yssO0wWomUyILqVeti
+PK+aXKwguI6bxLGZ3of0UH+mGsSl0mkp7kYZCm..OTQtfeRqP8rDSC7DgA
kHc5ajYqh04AzNFaxjRo+M3IGICUaOdKnXd0Fda..QwfoaX4QlRTgLqb7AN
ZTzM9WbmnyoXrx17kZlT3lsCgYEAm757XI3WJVj..WoLj1+v48WyoXZpcai
uv9bT4Cj+1XRS+gdKHK+SH7J3x2CRHVS+WH/SVC..DxuybvebDoT0TkKiCj
BWQaGzCaJqZa+POHK0klvS+9ln0/6k539p95tfX..X4TCzbVG6+gJiX0ysz
Yfehn5MCgYEAKMiKuWHCsVyCab3RUf6XA9gd3qY..fCTIGtSltr5PgFIV+G
engiVoWc/hkj8SBHzz1n1LN7KdF8ySU06MDggB..hJ+gXJKy+gf3mF5KmJ
DtKpjGHQzPF6vOe907y5NQLvVFGXuq/FIJZxB8k..fJdHEm2M4=
-----END RSA PRIVATE KEY-----
```

- **SSL key password:** enter the SSL password or passphrase for the SSL key. This field is masked.

Configure Intel 471 IOC transport and content types

Set up and configure transport and content types for Intel 471 IOC incoming feeds to retrieve and process indicators with bundled observables about compromised IP addresses, domains, URLs, and emails, with a focus on threat actors and groups.

To configure the general options for the Intel 471 IOC incoming feed, see [Configure incoming feeds](#).

About Intel 471 IOC

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Intel 471 IOC Feed	The channel provides data on compromised IP addresses, domains, URLs, emails, and actors.	Indicators of compromise with information on compromised domains, URLs, threat actors and groups.

**Warning:**

You should set the Intel 471 feed **Override TLP** option to **Amber**, since this content should not be redistributed outside the organization.

Configure transport and content types

Transport type	Allowed content types
Intel 471 IOC Feed	Intel 471

- From the **Transport type** drop-down menu, select **Intel 471 IOC Feed**.
- From the **Content type** drop-down menu, select **Intel IOC 471**.
The **Intel 471 IOC Feed** transport type supports only the **Intel IOC 471** content type.

The intel provider for the feed is Intel 471.

Under **Transport configuration**, configure the following settings:

- **API URL**: the URL pointing to the API endpoint exposing the service that makes the data available for retrieval through the feed. Contact the intel provider of the incoming feed to obtain this information.
The current API endpoint exposing the Intel 471 IOC Feed service is `https://api.intel471.com/v1/`
- **API key**: contact Intel 471 to receive an API key, and then enter it in the corresponding input field.
- **Email**: a valid email address to be granted access to the Intel 471 API endpoint to fetch the incoming feed from.

- **Polling start time**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days.

By default, the following checkboxes are preselected to include in the feed these indicators of compromise:

- **IOC type: IPAddress**: includes IOCs/observables describing compromised IP addresses.
- **IOC type: MaliciousURLs**: includes IOCs/observables describing compromised/malicious URLs.
- **IOC type: MD5**: includes IOCs/observables describing compromised MD5 file hashes.
- **IOC type: SHA256**: includes IOCs/observables describing compromised SHA-256 file hashes.

Configure Intel 471 Reports transport and content types

Set up and configure transport and content types for Intel 471 Reports incoming feeds to retrieve and process reports about compromised IP addresses, domains, URLs, and emails, with a focus on threat actors and groups.

To configure the general options for the Intel 471 Reports incoming feed, see [Configure incoming feeds](#).

About Intel 471 Reports

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Intel 471 Reports Feed	The channel provides data on compromised IP addresses, domains, URLs, emails, and actors.	Reports containing references to the information source, any additional comments by researchers or threat analysts, and any raw text content. Compromised domains, URLs, actors and so on are included in the report as observables, and they are linked to the parent entities they refer to.

**Warning:**

You should set the Intel 471 feed **Override TLP** option to **Amber**, since this content should not be redistributed outside the organization.

Configure transport and content types

Transport type	Allowed content types
Intel 471 Reports Feed	Intel 471

- From the **Transport type** drop-down menu, select **Intel 471 Reports Feed**.
- From the **Content type** drop-down menu, select **Intel 471**.
The **Intel 471 Reports Feed** transport type supports only the **Intel 471** content type.

The intel provider for the feed is Intel 471.

Under **Transport configuration**, configure the following settings:

- **API URL**: the URL pointing to the API endpoint exposing the service that makes the data available for retrieval through the feed. Contact the intel provider of the incoming feed to obtain this information.

The current API endpoint exposing the Intel 471 Reports Feed service is <https://api.intel471.com/v1/>
- **API key**: contact Intel 471 to receive an API key, and then enter it in the corresponding input field.

- **Email:** a valid email address to be granted access to the Intel 471 API endpoint to fetch the incoming feed from.
- **Polling start time:** select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days.

Configure MISP transport and content

Set up and configure transport and content types for MISP incoming feeds to retrieve and process information on indicators of compromise and MISP events retrieved from a source MISP instance.

To configure the general options for the MISP API incoming feed, see [Configure incoming feeds](#).

About MISP API

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
MISP API	MISP events and IOCs. Currently supported: <i>domain, hostname, ip-dst, ip-src, malware-sample, mutex, link, url, attachment, email-attachment, email-dst, email_src, target-email, filename, filename md5, filename sha1, filename sha256, md5, sha1, sha256, sha512, other</i>	Platform entities whose type is based on the ingested MISP data: campaigns, exploit targets, incidents, indicators, reports, threat actors, TTPs. Where applicable, ingested entities are automatically linked to show existing relationships.

Limitations

The native ingestion of MISP events is a significant improvement with respect to STIX-based MISP object ingestion. The current approach provides more context, and it produces a proper data hierarchy in the platform.

Keep in mind that there is a likely risk of not receiving all information, or of missing updates, due to limitations in this integration. Future platform releases will address these constraints. It is very likely that upon installing an upgraded release of the MISP integration, it will be necessary to recreate the data feed.

The following list includes all known limitations related to the current implementation.

- **Not all MISP properties** (<http://www.misp-project.org/datamodels/#types>) **and objects** (<http://www.misp-project.org/objects.html>) have been mapped to corresponding entity and observable data types in the platform.

At the moment, and after consulting with MISP Project team members, we implemented the most commonly used constructs.

- It is not yet possible to share MISP data to other MISP instances.

MISP allows data sharing across MISP instances, where events can evolve independently of the original source. When evolved events with same parent events are ingested from more than one source, the platform deduplicates them, and it ingests only the latest version, without performing any data merging.

- It is not yet possible to configure multiple MISP data sources.
- No special handling in place for proposals.

Configure transport and content types

Transport type	Allowed content types
MISP API	MISP JSON

- From the **Transport type** drop-down menu, select **MISP API**.
- From the **Content type** drop-down menu, select **MISP JSON**.
The **MISP API** transport type supports only the **MISP JSON** content type.

The intel provider for the feed is MISP.

Under **Transport configuration**, configure the following settings:

- **MISP URL**: the URL pointing to the API endpoint exposing the service that grants access to the source MISP instance.
- **MISP key**: contact the administrator of the source MISP instance to request an authentication MISP API key, and then enter it in the corresponding input field.
- **Polling start time**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days.

About MISP processing

MISP (<https://www.misp-project.org/>) is a decentralized, community-driven project. As such, instead of one central MISP data source, the effort relies on a collaborative network running multiple separate MIPS instances.

MISP makes it easier to gather, categorize, examine, and share indicators of compromise.

The online MISP platform structures its threat content by implementing **types** (<http://www.misp-project.org/datamodels/#types>), **attributes** (<https://www.misp-project.org/datamodels/#misp-default-attributes-and-categories>), **categories** (<https://www.misp-project.org/datamodels/#categories>), **taxonomies** (<https://www.misp-project.org/datamodels/#misp-taxonomies>), **objects** (<https://www.misp-project.org/datamodels/#misp-objects>), and **galaxies** (<https://www.misp-project.org/datamodels/#misp-galaxy>).

When the platform ingests data from a MISP instance through the MISP API / MISP JSON incoming feed, it analyzes and processes the source data to generate different entity types, based on the retrieved MISP content.

MISP attributes are processed as observables, and they are bundled with the entities they belong to.

When an ingested MISP object includes `admiralty-scale`, `kill-chain` and/or `tlp taxonomy` (<https://www.misp-project.org/taxonomies.html>) entries, the platform maps their values to its internal data schema accordingly.

Resulting entities

When MISP elements are part of a **galaxy** (<http://www.misp-project.org/galaxy.html>), they are ingested as entities whose type depends on the origin galaxy type:

Galaxy type	Resulting entity type
Attack Pattern, Banker, Exploit Kit, Intrusion Set, Malware, Ransomware, RAT, TDS, Tool	TTP
Microsoft Activity Group Actor, Threat- actor	Threat actor
Course of Action, Preventive Measure	Course of action

Incoming MISP events and objects are mapped and transformed to corresponding platform entities, based on type, attributes, taxonomy tags, and so on.

Indicator

This is the default entity resulting from a standard MISP event.

Exploit target

The ingested MISP object contains a `vulnerability` attribute or type, and a **CVE** (<https://www.cvedetails.com/>) attribute with a CVE ID.

The retrieved CVE ID is stored as a **Vulnerability** characteristic in the resulting exploit target entity.

Threat actor

The ingested MISP object contains a `threat-actor` attribute or type.

The source data is mapped to the corresponding fields in the resulting threat actor entity.

Report

The ingested MISP object holds data about an existing *threat actor* and an existing *exploit target*, it is categorized as *External analysis*, it is tagged as **OSINT** (https://www.misp-project.org/taxonomies.html#_osint), and its body is a piece of unstructured intelligence such as a blog post, a report, a presentation, and so on.

The resulting report entity includes relationships with the existing *threat actor* and *exploit target*.

If the source MISP object contains also a `link` attribute or type, the corresponding value is parsed to the report entity based on the value format:

- If the link value is text, it is parsed to the report entity in the *Description* section.
- If the link value is a URL pointing to a an asset or a resource, it is parsed to the report entity in the *Description* section.
If the URL link includes a text comment, the latter is parsed to the report entity in the *Description* section.

TTP

The ingested MISP object holds data about an existing *threat actor* and an existing *exploit target*.
The resulting TTP entity includes relationships with the existing *threat actor* and *exploit target*.

The ingested MISP object describes an incident, and it includes one of the following tags:

- `copyright-issue`
- `denial-of-service`
- `fastflux`
- `information-leak`

- malware
- phishing
- scam
- scan
- spam
- sql-injection
- system-compromise
- XSS

The resulting TTP entity includes relationships with the relevant indicator(s).

Campaign

The ingested MISP object contains a `campaign-name` and/or a `campaign-id` attribute or type.

If the source MISP object includes data about indicators, threat actors, and/or exploit targets, the resulting campaign entity includes relationships with the relevant indicator(s), as well as with the applicable threat actors and/or TTPs.

Configure OpenPhish transport and content

Set up and configure transport and content types for OpenPhish incoming feeds to ingest information about phishing sites and compromised URLs.

To configure the general options for the OpenPhish incoming feed, see [Configure incoming feeds](#).

About OpenPhish

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
OpenPhish Feed	Ingests the freely available OpenPhish Community feed, a list of phishing and compromised URLs.	URLs are saved as indicators. The signaled phishing activities are saved as TTPs related to the corresponding indicators.

Configure transport and content types

Transport type	Allowed content types
OpenPhish Feed	OpenPhish Feed Text

- From the **Transport type** drop-down menu, select **OpenPhish Feed**.
- From the **Content type** drop-down menu, select **OpenPhish Feed Text**.
The **OpenPhish Feed** transport type supports only the **OpenPhish Feed Text** content type.
- **OpenPhish API URL** : the URL pointing to the API endpoint exposing the service that grants access to the incoming feed data. Contact the intel provider to subscribe to the service and to obtain this information.
To retrieve the **Community** (https://openphish.com/phishing_feeds.html) feed, this field should be pre-populated with <https://openphish.com/feed.txt>.

Configure PhishMe Intelligence transport and content

Set up and configure transport and content types for PhishMe Intelligence incoming feeds to retrieve and process PhishMe intelligence reports on malware and phishing campaigns.

To configure the general options for the PhishMe Intelligence incoming feed, see [Configure incoming feeds](#).

About PhishMe Intelligence

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
PhishMe Intelligence	PhishMe Intelligence reports published since a specific start date.	STIX reports focusing on malware and phishing campaigns.

Configure transport and content types

Transport type	Allowed content types
PhishMe Intelligence API	STIX 1.1.1

- From the **Transport type** drop-down menu, select **PhishMe Intelligence API**.
- From the **Content type** drop-down menu, select **STIX 1.1.1**.
The **PhishMe Intelligence API** transport type supports only the **STIX 1.1.1** content type, which is essentially standard STIX 1.1.1.

The organization providing the source data for the incoming feed is PhishMe.

Under **Transport configuration**, configure the following settings:

- **PhishMe Intelligence API URL**: the URL pointing to the API endpoint exposing the service that grants access to the incoming feed data. Contact the intel provider to subscribe to the service and to obtain this information.

The current API endpoint exposing the PhishMe Intelligence service is <https://www.threathq.com>
- **API Username**: a valid user name to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
- **API Password**: a valid password to authenticate and be granted the necessary authorization to access the data source and to download/ingest data.
Contact the intelligence provider to subscribe to the service and to obtain the required authentication and authorization credentials.
- **Ingest feed starting from**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 365 days/one year.

Configure Proofpoint Email Brand Defense transport and content

Set up and configure transport and content types for Proofpoint Email Brand Defense incoming feeds to retrieve and process information on email threats such as phishing, spoofing, and email malware.

To configure the general options for the Proofpoint Email Brand Defense incoming feed, see [Configure incoming feeds](#).

About Proofpoint Email Brand Defense

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Proofpoint Email Brand Defense	Domain names, IP addresses, IDS rules (firewall, blocklists), Snort (Suricata-compatible).	Indicators and observables focusing on email threats such as phishing, spoofing, email malware, and impostor email/fraudulent messages.

Configure transport and content types

Transport type	Allowed content types
Proofpoint Email Brand Defense threat API	Proofpoint Message

- From the **Transport type** drop-down menu, select **Proofpoint Email Brand Defense threat API**.
- From the **Content type** drop-down menu, select **Proofpoint Message**.
The **Proofpoint Email Brand Defense** transport type supports only the **Proofpoint Message** content type.

Under **Transport configuration**, configure the following settings:

- **API key**: contact Proofpoint to receive an API key, and then enter it in the corresponding input field.
- **API URL**: the URL pointing to the API endpoint exposing the service that makes the data available for retrieval through the feed. Contact the intel provider of the incoming feed to obtain this information.

The current API endpoint exposing the Proofpoint Email Brand Defense threat API service is <https://api.emaildefense.proofpoint.com/v1>

- **Likely impact lower threshold:** enter an integer value to assess the maliciousness confidence level of processed email threats.
This value sets the minimum maliciousness confidence value to flag a potential email threat as *somewhat likely* to be malicious.
The lower threshold value needs to be smaller than the highest threshold value.
 - Enter a value between 0 and 100.
 - Default value: 60.
- **Likely impact higher threshold:** enter an integer value to assess the maliciousness confidence level of processed email threats.
This value sets the minimum maliciousness confidence value to flag a potential email threat as *very likely* to be malicious.
The higher threshold value needs to be bigger than the lower threshold value.
 - Enter a value between 0 and 100.
 - Default value: 90.
- **Polling start time:** select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days. - **Domain:** enter a domain name to retrieve information on fraudulent email messages sent on behalf of the specified domain.

Returned data includes, when available, the brand a spoofing message uses to trick users, the message subject line, source IP address, source host name, and source email address, as well as any MD5 file hashes, if the malicious email contains attachments.

If you do not specify any domain name, the feed defaults to retrieving a list with the most relevant threats, based on your Proofpoint Email Fraud Defense subscription and account preferences. The list contains the most relevant fraudulent messages reported in the previous seven days.

About impact thresholds

- Ingested observables with a *lower* value than the one you set in **Likely impact lower threshold** are automatically flagged as **Malicious - Low confidence**.
- Ingested observables whose likely impact value is *bigger* than the *lower threshold* and *smaller* than the *higher threshold* values are automatically flagged as **Malicious - Medium confidence**.
- Ingested observables with a *higher* value than the one you set in **Likely impact higher threshold** are automatically flagged as **Malicious - High confidence**.

You can use impact threshold values to automatically filter enricher output data, and to instrument external systems and devices, such as a SIEM, to execute follow-up actions. For example, add all high confidence maliciousness email addresses to a blocklist, or route all ingested Snort rules with a high maliciousness confidence to a Snort instance to trigger rule-driven actions.

Configure SpyCloud Breach API transport and content

Set up and configure transport and content types for SpyCloud Breach API incoming feeds to retrieve and process information on incidents, security breaches, and account takeovers (ATO).

To configure the general options for the SpyCloud Breach API incoming feed, see [Configure incoming feeds](#).

About SpyCloud Breach API

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
SpyCloud Breach API	Incident and breach data, along with relevant context.	Incident entities focusing on security breaches and account takeovers, CIQ entities, CybOX observables, related observables. Where available, context data include threat actor, targeted victim, affected assets, and geolocation details.

Configure transport and content types

Transport type	Allowed content types
SpyCloud Breach API	SpyCloud JSON

- From the **Transport type** drop-down menu, select **SpyCloud Breach API**.
- From the **Content type** drop-down menu, select **SpyCloud Message**.
The **SpyCloud Breach API** transport type supports only the **SpyCloud Message** content type.

The organization providing the source data for the incoming feed is SpyCloud.

Under **Transport configuration**, configure the following settings:

- **SpyCloud API key**: contact SpyCloud and create a user account to receive an API key; then enter it in the corresponding input field.
- **Polling start time**: select an initial date if you want to fetch content from the intel provider/data source starting from a specific date in the past.

By default, the max. amount of days in the past per each query/request is set to 60 days.

Configure Threat Recon transport and content

Set up and configure transport and content types for Threat Recon incoming feeds to retrieve and process indicators with information on compromised and malicious IP addresses, domains, as well as whois information.

To configure the general options for the Threat Recon JSON API incoming feed, see [Configure incoming feeds](#).

About Threat Recon JSON API

This intel provider/intel source enables intelligence ingestion through the following channels:

Feed	Ingested data	Processed data
Threat Recon JSON API	The channel provides data on compromised IP addresses, domains, as well as whois information.	Indicators with embedded observables, where each observable represents an indicator of compromise (IOC).

Configure transport and content types

Transport type	Allowed content types
Threat Recon JSON API	Threat Recon

- From the **Transport type** drop-down menu, select **TAXII poll**.
- From the **Content type** drop-down menu, select **STIX 1.1.1**.

The intel provider for the feed is Threat Recon.

Under **Transport configuration**, configure the following settings:

- **API key**: contact Threat Recon to receive an API key, and then enter it in the corresponding input field.

Incoming feeds reference

Reference section with lookup information on supported incoming feeds, content types, and transport types.

Available incoming feeds

The overview lists and points to the articles on the available incoming feeds. Each article describes how to configure the specific options for each incoming feed.

Typically, incoming feeds use different *transport types* and *content types*. The other configuration options are identical across all incoming feeds.

Title	Excerpt
Configure Amazon S3 poll transport and content	Set up and configure transport and content types for Amazon S3 poll incoming feeds to securely transfer data from selected Amazon S3 buckets.
Configure Anubis Cyberfeed transport and content	Set up and configure transport and content types for AnubisNetworks Cyberfeed incoming feeds to retrieve and process information on compromised machines and DNS servers, as well as compromised web ...
Configure BFK transport and content	Set up and configure transport and content types for BFK incoming feeds to retrieve and process reports on cyber threats and activities, as well as information on NIDs (Network Intrusion Detections).
Configure Crowdstrike Falcon Intelligence Indicator transport and content	Set up and configure transport and content types for Crowdstrike Falcon Intelligence Indicator incoming feeds to retrieve and process information on indicators, such as compromised devices, malicio...
Configure Crowdstrike Falcon Intelligence Reports transport and content	Set up and configure transport and content types for Crowdstrike Falcon Intelligence Report incoming feeds to retrieve and process Crowdstrike intelligence reports, periodic reports, and tippers.
Configure Crowdstrike Falcon Intelligence Threat Actor transport and content	Set up and configure transport and content types for Crowdstrike Falcon Intelligence Threat Actor incoming feeds to retrieve and process intelligence on threat actors, their TTPs, and any campaigns...
Configure CVE Search API transport and content	Set up and configure transport and content types for CVE Search API incoming feeds to retrieve and process information on common software and hardware vulnerabilities, along with the corresponding ...
Configure IMAP email fetcher transport and content	Set up and configure transport and content types for IMAP email fetcher incoming feeds to retrieve and process information from email body contents.

Title	Excerpt
Configure IMAP email attachment fetcher transport and content	Set up and configure transport and content types for IMAP email attachment fetcher incoming feeds to retrieve email attachment content in multiple formats. You can then use the resulting reports fo...
Configure FireEye iSIGHT Intelligence Report API transport and content	Set up and configure transport and content types for FireEye iSIGHT Intelligence Report API incoming feeds to retrieve and process STIX reports on vulnerabilities, malware, and other threats such a...
Configure FTP download transport and content	Set up and configure transport and content types for FTP download incoming feeds to retrieve and process information from specific data sources supporting the FTP protocol.
Configure Group-IB transport and content	Set up and configure transport and content types for Group-IB incoming feeds to retrieve and process information on compromised logins and (email) accounts, payment cards, mobile devices, money mul...
Configure HTTP download transport and content	Set up and configure transport and content types for HTTP download incoming feeds to retrieve and process information from specific data sources supporting the HTTP protocol.
Configure Intel 471 Reports transport and content types	Set up and configure transport and content types for Intel 471 Reports incoming feeds to retrieve and process reports about compromised IP addresses, domains, URLs, and emails, with a focus on thre...
Configure Intel 471 IOC transport and content types	Set up and configure transport and content types for Intel 471 IOC incoming feeds to retrieve and process indicators with bundled observables about compromised IP addresses, domains, URLs, and emai...
Configure MISP transport and content	Set up and configure transport and content types for MISP incoming feeds to retrieve and process information on indicators of compromise and MISP events retrieved form a source MISP instance.
Configure mount point download transport and content	Set up and configure transport and content types for mount point download incoming feeds to retrieve and process information from specific data sources available on the (local) network.
Configure NSFocus transport and content	Set up and configure transport and content types for NSFocus incoming feeds to retrieve and ingest NSFocus security events reporting incidents and security breaches.
Configure OpenPhish transport and content	Set up and configure transport and content types for OpenPhish incoming feeds to ingest information about phishing sites and compromised URLs.
Configure PhishMe Intelligence transport and content	Set up and configure transport and content types for PhishMe Intelligence incoming feeds to retrieve and process PhishMe intelligence reports on malware and phishing campaigns.
Configure Proofpoint Email Brand Defense transport and content	Set up and configure transport and content types for Proofpoint Email Brand Defense incoming feeds to retrieve and process information on email threats such as phishing, spoofing, and email malware.
Configure SFTP download transport and content	Set up and configure transport and content types for SFTP download incoming feeds to securely transfer data from selected SFTP servers.

Title	Excerpt
Configure SpyCloud Breach API transport and content	Set up and configure transport and content types for SpyCloud Breach API incoming feeds to retrieve and process information on incidents, security breaches, and account takeovers (ATO).
Configure TAXII inbox transport and content	Set up and configure transport and content types for TAXII inbox incoming feeds to retrieve and process information from specific data sources supporting the TAXII transport type.
Configure TAXII poll transport and content	Set up and configure transport and content types for TAXII poll incoming feeds to poll discovery services to retrieve and process information from specific data sources supporting the TAXII transport...
Configure Cisco Threat Grid Samples API transport and content	Set up and configure transport and content types for Cisco Threat Grid Samples API incoming feeds to retrieve and process information on compromised IP addresses, domains, hashes, registry keys, an...
Configure Cisco Threat Grid Curated Feed transport and content	Set up and configure transport and content types for Cisco Threat Grid Curated Feed incoming feeds to retrieve and process information on Trojan-infected machines and servers, on malware-infected D...
Configure Threat Recon transport and content	Set up and configure transport and content types for Threat Recon incoming feeds to retrieve and process indicators with information on compromised and malicious IP addresses, domains, as well as w...
Exchange data between platforms	Configure TAXII feeds to enable data exchange between two platform instances.
About ingestion	EclecticIQ Platform can ingest data in different formats from multiple sources. Before storing ingested data as entities and observables, the input data is deduplicated, normalized, and processes t...

Content types

These are the data formats the platform can process through feeds.

Under *Feed type* **in** defines an input format that incoming feeds ingest; **out** defines an output format that outgoing feeds publish.

Content type	Feed type	Description
Anubis Cyberfeed JSON	in	JSON format representing entity data as JSON objects.
ArcSight CEF	out	The Common Event Format is a text-based standard for log records proposed by ArcSight. It allows sharing, consuming, and parsing event information across devices such as SIEM platforms and Syslog servers.
Cisco Threat Grid Samples JSON	in	JSON format representing entity data as JSON objects.

Content type	Feed type	Description
EclecticIQ Entities CSV	out	Comma separated CSV format for tabular data representation of entities.
EclecticIQ Observables CSV	out	Comma separated CSV format for tabular data representation of observables.
EclecticIQ HTML Report	out	Default HTML format to publish EclecticIQ intel reports.
EclecticIQ HTML Report Digest	out	Default HTML format to publish EclecticIQ intel report digests.
EclecticIQ JSON	in, out	JSON format representing entity data as JSON objects.
Intel 471	in	Intel 471 reports. Bundled observables are linked to the parent report entity. API endpoint: https://api.intel471.com/v1/reports/{}
PDF	in, out	Standard PDF format, preferably native (not scanned).
STIX 1.0	in, out	STIX data model v. 1.0 (http://stixproject.github.io/data-model/1.0/).
STIX 1.1	in, out	STIX data model v. 1.1 (http://stixproject.github.io/data-model/1.1/).
STIX 1.1.1	in, out	STIX data model v. 1.1.1 (http://stixproject.github.io/data-model/1.1.1/).
STIX 1.2	in, out	STIX data model v. 1.2 (http://stixproject.github.io/data-model/1.2/).
Text/Plain text value	in, out	Plain text format. This content type allows you to enter free text and literals, wildcards (where supported), as well as JSON paths to point to specific entity property fields, and regex patterns to filter data.
Threat Recon	in	Threat Recon JSON output returned by the Threat Recon API (https://threatrecon.co/api). Threat Recon focuses on providing information about indicators.
STIX 1.1.1	in	FireEye iSIGHT Intelligence Report API outputs reports in STIX 1.1.1 format. Reports concern threat topics such as vulnerabilities, malware, threat actors, strategies, tactics, and techniques.
BFK Threat Intelligence JSON	in	BFK reports and NIDs (Network Intrusion Detections) are saved as JSON report entities; they concern threat topics such as threat actors, targeted victims, tactics, and techniques.
Crowdstrike indicator JSON	in	Indicators retrieved from the Falcon Intelligence platform such as compromised devices, malicious domains, hashes, and so on starting from the specified polling date.
CAPEC XML	in, out	Categorized and enumerated attack patterns, attack mechanisms, strategies, tactics and techniques retrieved from the CAPEC (https://capec.mitre.org/about/index.html) catalog.

Content type	Feed type	Description
CrowdStrike report JSON	in	Reports retrieved from the Falcon Intelligence platform in JSON format and as PDF attachments.
CrowdStrike actor JSON	in	Threat actor entities, related TTPs, indicators, and campaigns, as well as related observables to represent actor ID, target country, target industry, and targeted victim(s).
CVE Search JSON	in	Exploit target entities retrieved from CIRCL CVE Search (https://www.circl.lu/services/cve-search/). The entity ID is derived from the CVE ID (https://cve.circl.lu/). API endpoint: https://cve.circl.lu/api/last .
Intel 471 IOC Feed	in	Indicators of compromise such as IP addresses, malicious URLs, and MD5 and SHA-256 hashes. Intel 471 focuses on providing first-hand information related to threat actors and groups. API endpoint: https://api.intel471.com/v1/search/ .
OpenPhish Feed Text	in	Phishing URLs are saved as indicators. The signaled phishing activities are saved as TTPs related to the corresponding indicators. API endpoint: https://openphish.com/feed.txt .
Proofpoint Message	in	Indicators and observables focusing on email threats such as phishing, spoofing, email malware, and impostor email/fraudulent messages API endpoint: https://api.emaildefense.proofpoint.com/v1 .

Transport types

These are the supported communications protocols the platform uses to ingest data through incoming feeds.

Transport type	Feed type	Description
Anubis Cyberfeed	in	Provides data on bank Trojans, compromised DNS servers, malware-infected web site and malware files.
Cisco Threat Grid Curated Feed	in	Provides data on compromised IP addresses, domains, hashes, registry keys, and network streams.
Cisco Threat Grid Samples API	in	Allows submitting malware samples for analysis, as well as investigating a domain, an IP, or a URL to obtain information about potential threats.
FTP download	in	Custom feed ingesting data through FTP.
TAXII poll for Group-IB	in	The Group-IB TAXII poll service offers a range of channels that provide information and comprehensive coverage on cyber threats and vulnerabilities such as compromised logins, passwords, corporate email accounts, compromised bank card numbers and online banking keys, compromised mobile devices, money mules, Tor nodes, and so on.
HTTP download	in	Custom feed ingesting data through HTTP.
IMAP email fetcher	in	Custom feed using the IMAP email protocol to ingest content included in email body sections .

Transport type	Feed type	Description
Intel 471 Reports Feed	in	Ingests Intel 471 reports on compromised IP addresses, domains, URLs, and emails, with a focus on threat actors and groups. API endpoint: https://api.intel471.com/v1/
Mount point download	in	Custom feed using a local or a network drive as a data source.
TAXII inbox	in	Custom feed ingesting data through the TAXII inbox service.
TAXII poll	in	Custom feed ingesting data through the TAXII poll service.
Threat Recon JSON API	in	Provides data on compromised IP addresses, domains, as well as whois information.
FireEye iSIGHT Intelligence Report API	in	Provides intelligence reports on vulnerabilities, malware, and other threats such as threat actors, strategies, tactics, and techniques.
IMAP email attachment fetcher	in	Ingests email attachments that are then transformed into reports. The reports can be used for analysis (human consumption) or instrumentation (machine consumption/automation).
BFK API	in	Ingested BFK reports are saved as report entities in the platform, whereas ingested NIDs (Network Intrusion Detections) result in entities with a linked TTP.
CrowdStrike Falcon Intelligence Indicator Feed	in	Ingests information about indicators by polling the Falcon Intel Indicator API. It is possible to set a start date to poll data from.
CrowdStrike Falcon Intelligence Reports Feed	in	Ingests CrowdStrike Intelligence Reports (CSIR), Threat Assessments (CSTA), Alerts (CSA), Periodic Reports (CSMR), Tippers (CSIT) in JSON format.
PhishMe Intelligence API	in	Ingests PhishMe reports focusing on malware and phishing campaigns.
CVE Search API	in	Ingests CVE information about common software and hardware vulnerabilities, along with the corresponding exposures. The enricher contacts the Computer Incident Response Center Luxembourg (CIRCL) cve-search API to retrieve all the available details associated with the input CVE IDs. Ingested data is stored in the platform as exploit target entities. The exploit target ID is derived from the corresponding source CVE.
SpyCloud Breach API	in	Ingests incident and security breach information from the SpyCloud breach catalog, along with context-relevant metadata such as affected sites, number of records ingested from the breach, if and when the stolen records were put for sale online, and so on. Searches for breach records matching predefined inputs such as domain name, IP address, or e-mail address. Ingested data is stored in the platform as incident entities, CIQ 3.0 (http://docs.oasis-open.org/ciq/v3.0/specs/ciq-specs-v3.html) entities, CybOX observables, and related observables.
Amazon S3 poll	in	Ingests retrieved data based on the configured content type for the feed. After successfully retrieving and ingesting the data, it is removed from the source Amazon S3 bucket (https://docs.aws.amazon.com/amazons3/latest/dev/usingbucket.html).

Transport type	Feed type	Description
Crowdstrike Falcon Intelligence Threat Actor Feed	in	Ingests Crowdstrike Falcon Intelligence information on threat actor entities, related TTPs, indicators, and campaigns, as well as related observables to represent actor ID, target country, target industry, and targeted victim(s).
OpenPhish Feed	in	Ingests the OpenPhish Community feed, a list of phishing URLs. Ingested data is stored in the platform as indicators — the retrieved phishing URLs — and related TTPs representing the corresponding phishing activities.
SFTP download	in	Ingests retrieved data from the configured SFTP server location, based on the configured content type for the feed. SFTP server URL format: <i>sftp://\${sftp_server}:\${port}/\${path_to_target_dir}</i>
Proofpoint Email Brand Defense threat API	in	Ingests indicators and observables focusing on email threats such as phishing, spoofing, email malware, and impostor email/fraudulent messages. API endpoint: <i>https://api.emaildefense.proofpoint.com/v1</i>
Intel 471 IOC Feed	in	Ingests Intel 471 indicators and bundled observables describing compromised IP addresses, domains, URLs, and emails, with a focus on threat actors and groups. API endpoint: <i>https://api.intel471.com/v1/</i>
MISP API	in	Ingests MISP events and IOCs and processes them to platform entities whose type — for examples, incident, threat actor, report, and so on — is mapped based on the ingested source MISP data.
NSFocus Provider	in	Ingests NSFocus security events and processes them as report entities. IP addresses, domain names, and file hashes referenced in the reports are saved as observables related to top-level incident entities. CVE vulnerabilities referenced in the reports are saved as observables related to top-level exploit target entities.

Configure enrichers

Configure enrichers to augment intelligence value with additional context obtained from selected intel providers and data sources.

**Danger:**

Due to breaking changes, the following enrichers and incoming feeds are *not working* on EclecticIQ Platform release 1.14.4 (and earlier) until release 2.0.2 included:

- Enrichers:
 - FireEye iSIGHT
 - Intel 471
 - Recorded Future
- Incoming feeds:
 - BFK API
 - Cisco Threat Grid Curated Feed
 - Cisco Threat Grid Samples API
 - Crowdstrike Falcon Intelligence Indicator Feed
 - Crowdstrike Falcon Intelligence Reports Feed
 - Crowdstrike Falcon Intelligence Threat Actor Feed
 - FireEye iSIGHT Intelligence Report API

To restore full functionality for these enrichers and incoming feeds, upgrade to EclecticIQ Platform release 2.1.0 or later.

Enrichers poll external data sources to provide additional context and detail to augment — hence, enrich — the intelligence value of entities and observables stored in the platform.

The platform ships with a broad selection of built-in, ready-to-use enrichers to obtain geolocation IP and whois details, DNS domain and malware information, as well as other relevant data to help analysts draw a sharper and more comprehensive picture of the cyber threat relationships and the cyber threat scenarios under investigation.

This article describes how to configure the general options for all enrichers. These options are identical for all standard platform enrichers.

Besides the general options, each enricher has specific configuration settings. For example, API endpoints, API keys, login credentials to access a subscription service or a data source, and so on.

The configuration options that are specific to each enricher are grouped together under the **Parameters** section on the enricher configuration page.

For the configuration details, follow the links under Configure specific enricher parameters.

Configure the general options

To configure or to edit an enricher task, do the following:

- On the top navigation bar click **Data configuration > Enrichers**.
- On the enricher overview page click the tile corresponding to the enricher you want to configure or modify.
- On the enricher detail page click the **Edit** button.

✓ Input fields marked with an asterisk are required.

- **Name**: the name used to identify the enricher. It should be descriptive and easy to remember.
- **Description**: additional textual details. If you want, you can add a short description to provide more information and context.
- **Cache validity (sec)**: defines for how long enrichment data remains stored in the cache. The value is expressed in seconds.
- **Rate limit (per sec)**: sets the maximum allowed number of requests/executions per second.
- **Monthly execution cap (executions)**: sets a maximum allowed number of requests/executions per month. Together with rate limiting, execution cap helps control data traffic for the enricher; for example, when the API or the service you are connecting to enforces usage limits.
- **Source reliability**: from the drop-down menu select an option to flag the feed or enricher content with a predefined reliability value to help other users assess how trustworthy the data source is.
Values in this menu have the same meaning as the first character in the **two-character Admiralty System code** (https://en.wikipedia.org/wiki/admiralty_code).
Example: *B - Usually reliable*
- **Enabled**: checkbox. Select the **Enabled** checkbox to enable the enricher task immediately after editing and saving it. If you select the checkbox, the enricher or the rule is executed automatically. If you deselect it, you need to manually run the enricher or the rule.

Under **Parameters**, define the specific configuration parameters for the selected enricher.

Edit enricher tasks

To configure or to edit an enricher task, do the following:

- On the top navigation bar click **Data configuration > Enrichers**.
- On the enricher overview page click the tile corresponding to the enricher you want to configure or modify.
- On the enricher detail page click the **Edit** button.

✓ Input fields marked with an asterisk are required.

- **Name**: the name used to identify the enricher. It should be descriptive and easy to remember.
- **Description**: additional textual details. If you want, you can add a short description to provide more information and context.
- **Cache validity (sec)**: defines for how long enrichment data remains stored in the cache. The value is expressed in seconds.
- **Rate limit (per sec)**: sets the maximum allowed number of requests/executions per second.

- **Monthly execution cap (executions)**: sets a maximum allowed number of requests/executions per month. Together with rate limiting, execution cap helps control data traffic for the enricher; for example, when the API or the service you are connecting to enforces usage limits.
- **Source reliability**: from the drop-down menu select an option to flag the feed or enricher content with a predefined reliability value to help other users assess how trustworthy the data source is. Values in this menu have the same meaning as the first character in the **two-character Admiralty System code** (https://en.wikipedia.org/wiki/admiralty_code).
Example: *B - Usually reliable*
- **Enabled**: checkbox. Select the **Enabled** checkbox to enable the enricher task immediately after editing and saving it. If you select the checkbox, the enricher or the rule is executed automatically. If you deselect it, you need to manually run the enricher or the rule.

Under **Parameters**, define the specific configuration parameters for the selected enricher.



Some enrichers may require a paid subscription to the data service provider to be granted access to the source data.

Contact the intel service provider whose data you want to use as a source for the enricher to request the necessary details, such as API endpoint or API key, as well as any required authentication and authorization credentials.

Configure enricher rules

Enrichment rules define what to do with the retrieved enrichment data.

Rules act like filters, and they set the logical constraints defining:

- The platform data sources to augment with enrichment information.
Data sources you can enrich are incoming feeds, other enrichers, and groups.
- Within the selected platform data sources, the entity type(s) to augment with enrichment information.
- The enrichers to use to fetch enrichment data.

Add enricher rules

To add a new enricher rule, do the following:

- On the top navigation bar click **Data configuration > Rules > Enrichment**.
- The **Rules > Enrichment** page shows an overview of the configured enricher rules. You can sort the items on the view by column header. To do so, click the column header you want to base the data sorting on. An upward-pointing ▲ or a downward-pointing ▼ arrow in the header indicates ascending and descending sort order, respectively.
- Click the + icon.



Input fields marked with an asterisk are required.

On the **Create enrichment rule** page, fill out the fields to create the new enricher rule:

- **Name:** define a name to identify the rule. It should be descriptive and easy to remember.
- **Description:** additional textual details. If you want, you can add a short description to provide more information and context.
- Click **+ Add** or **+ More** to add a filtering option.
- **Source:** from the drop-down menu select the incoming feed, enricher, or group whose entities and observables you want to augment with additional information.
- **Entity types:** from the drop-down menu select the entity types you want to enrich with additional information.
- **TLP:** from the drop-down menu select the TLP color code you want to use to filter enrichment data.
TLP (<https://www.us-cert.gov/tlp>) provides an intuitive reference to assess how sensitive information is, focusing in particular on how serious it is, and whom it should or should not be shared with.
- Click **+ Add** or **+ More** to add a new filtering option. For example, to include another incoming feed or a different entity type. A filter can take only one source and one entity type at a time, but you can set up rules with as many filters as you need.
- **Enrichers:** from the drop-down menu select one or more enrichers to apply the rule to.
When a rule is applied to one or more enrichers, it filters the enrichment data polled from the enricher source, based on the specified rule filters and criteria.
- Select the **Enabled** checkbox to enable the rule immediately after creating it.
- Click **Save** to store your changes, or **Cancel** to discard them.

Save options

Besides committing the current data by clicking **Save**, you can also click the downward-pointing arrow on the **Save** button to display a context menu with additional save options:

- **Save and new:** saves the current data for the active item, and it allows you to start creating a new item of the same type right away. For example, a dataset, a feed, a rule, a workspace, or a task.
- **Save and duplicate:** saves the current data for the active item, and it creates a pre-populated copy of the same item, which you can use as a template to speed up manual work.

Edit enricher rules

To edit enricher rules, do the following:

- On the top navigation bar click **Data configuration > Rules > Enrichment**.
- The **Rules > Enrichment** page shows an overview of the configured enricher rules.
You can sort the items on the view by column header. To do so, click the column header you want to base the data sorting on. An upward-pointing ▲ or a downward-pointing ▼ arrow in the header indicates ascending and descending sort order, respectively.

To edit the details of a specific rule, do the following:

- Click an area on the row corresponding to the rule you want to examine. An overlay slides in from the side of the screen to display the rule detail pane.
- On the rule detail pane click **Actions > Edit**.

Alternatively:

- Click the ⋮ icon on the row corresponding to the enricher you want to configure or modify.

- From the drop-down menu select **Edit**.

✓ Input fields marked with an asterisk are required.

- **Name**: define a name to identify the rule. It should be descriptive and easy to remember.
- **Description**: additional textual details. If you want, you can add a short description to provide more information and context.
- **Source**: from the drop-down menu select the incoming feed, enricher, or group whose entities and observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity types you want to enrich with additional information.
- **TLP**: from the drop-down menu select the TLP color code you want to use to filter enrichment data.
TLP (<https://www.us-cert.gov/tlp>) provides an intuitive reference to assess how sensitive information is, focusing in particular on how serious it is, and whom it should or should not be shared with.
- Click **+ Add** or **+ More** to add a new filtering option. For example, to include another incoming feed or a different entity type.
- **Enrichers**: from the drop-down menu select one or more enrichers to apply the rule to. They are external data providers that are polled to obtain relevant enricher raw data; for example, whois lookup, reverse DNS, or GeoIP information.
- Select the **Enabled** checkbox to enable the rule immediately after creating it.
- Click **Save** to store your changes, or **Cancel** to discard them.

Delete enricher rules


To delete an enricher rule, do the following:

- On the top navigation bar click **Data configuration > Rules > Enrichment**.
- The **Rules > Enrichment** page shows an overview of the configured enricher rules.
You can sort the items on the view by column header. To do so, click the column header you want to base the data sorting on. An upward-pointing ▲ or a downward-pointing ▼ arrow in the header indicates ascending and descending sort order, respectively.

To delete a specific rule, do the following:

- Click an area on the row corresponding to the rule you want to delete. An overlay slides in from the side of the screen to display the rule detail pane.
- On the rule detail pane click **Actions > Delete**.

Alternatively:

- Click the  icon on the row corresponding to the rule you want to delete.
- From the drop-down menu select **Delete**.
- On the confirmation pop-up dialog, click **Delete** to confirm the action.
- The rule is deleted.

Configure specific enricher parameters

- Censys enricher
- CIRCL IPs related to SSL certificate enricher
- CIRCL SSL Certificate Fetcher enricher
- Cisco Threat Grid enricher
- Crowdstrike Falcon Intelligence Indicator enricher
- CVE Search enricher
- DomainTools Hosted Domains enricher
- DomainTools Malicious Server Domains enricher
- DomainTools Parsed Whois enricher
- DomainTools Reputation enricher
- DomainTools Reverse Whois enricher
- DomainTools Suspicious Domains enricher
- Elasticsearch sightings enricher
- Create custom enrichers
- Farsight DNSDB enricher
- FireEye iSIGHT enricher
- Flashpoint AggregINT enricher
- Flashpoint Blueprint enricher
- Flashpoint Thresher enricher
- Fox-IT InTELL Portal enricher
- Intel 471 enricher
- MaxMind GeoIP enricher
- MISP API enricher
- NSFocus Intelligence enricher
- OpenDNS OpenResolve enricher
- PassiveTotal Passive DNS enricher
- PassiveTotal IP/Domain enricher
- PassiveTotal Malware enricher
- PassiveTotal Whois enricher
- Proofpoint Email Threat enricher
- PyDat enricher
- Recorded Future enricher
- RIPEstat GeoIP enricher
- RIPEstat Whois enricher

- Shodan enricher
- Splunk sightings enricher
- SpyCloud Breach Data enricher
- ThreatCrowd enricher
- Unshorten-URL enricher
- VirusTotal enricher

Enable and disable enrichers

Enable and disable enrichers to start and stop enrichment tasks as needed.

View enricher tasks

To view enricher tasks, do the following:

- On the top navigation bar click **Data configuration > Enrichers**.
- On the enricher overview page click the enricher you want to examine.
- On the enricher detail page, you can view all the details about the selected enricher, including the rules driving the enricher behavior, recently executed enriching tasks, and the state.
- You can click the state value or an enrichment rule to display additional information.



When the state value returns **FAILURE**, click the link to view the task execution traceback and to begin troubleshooting.

The **Data configuration > Enrichers** view shows all configured enrichers polling third-party and/or external services to acquire additional information to integrate observables with, so that they can provide more context to the cyber threat entities they belong to.

Data Configuration	Incoming feeds	Outgoing feeds	Taxonomies	Enrichers	Rules
<input checked="" type="checkbox"/> Enabled 58 runs this month	<input checked="" type="checkbox"/> Enabled 11 runs this month	<input type="checkbox"/> Enabled 18 runs this month	<input checked="" type="checkbox"/> Enabled 48 runs this month	<input type="checkbox"/> Enabled 1 runs this month	<input checked="" type="checkbox"/> Enabled 15 runs this month
Elastic Sightings Enricher <input checked="" type="checkbox"/> Enabled 217 runs this month	Farsight DNSDB <input type="checkbox"/> Enabled 26 runs this month	FireEye <input type="checkbox"/> Enabled 148 runs this month	Flashpoint AggregINT Enricher <input checked="" type="checkbox"/> Enabled 109 runs this month	Flashpoint Blueprint Enricher <input checked="" type="checkbox"/> Enabled 158 runs this month	Flashpoint Thresher Enricher <input type="checkbox"/> Enabled 50 runs this month
Fox-IT INTELL Portal <input type="checkbox"/> Enabled 51 runs this month	Intel 471 <input checked="" type="checkbox"/> Enabled 103 runs this month	OpenResolve <input checked="" type="checkbox"/> Enabled 81 runs this month	PassiveTotal IP/Domain Enricher <input checked="" type="checkbox"/> Enabled 60 runs this month	PassiveTotal Malware Enricher <input checked="" type="checkbox"/> Enabled 53 runs this month	PassiveTotal Passive DNS Enricher <input type="checkbox"/> Enabled 48 runs this month
PassiveTotal Whois Enricher <input type="checkbox"/> Enabled 43 runs this month	PyDat <input type="checkbox"/> Enabled 23 runs this month	RIPeStat GeoIP <input checked="" type="checkbox"/> Enabled 20 runs this month	RIPeStat Whois <input checked="" type="checkbox"/> Enabled 18 runs this month	Recorded Future <input type="checkbox"/> Enabled 121 runs this month	Splunk Sightings Enricher <input type="checkbox"/> Enabled 173 runs this month
Threatcrowd API V2 <input checked="" type="checkbox"/> Enabled 120 runs this month	Unshorten-URL enricher <input checked="" type="checkbox"/> Enabled 41 runs this month	VirusTotal <input type="checkbox"/> Enabled 100 runs this month			

Enable and disable enrichers

To enable an enricher so that it runs automatically, do the following:

- On the top navigation bar click **Data configuration > Enrichers**.
- On the enricher overview page browse to the enricher you want to enable.
- Select the **Enabled** checkbox to enable the enricher so that it runs automatically.

The screenshot shows the 'Data Configuration' menu with 'Enrichers' selected. The page displays three enricher cards:

Enricher Name	Status	Runs this month
CVE Search enricher	<input checked="" type="checkbox"/> Enabled	0
Censys Enricher	<input checked="" type="checkbox"/> Enabled	59
Cisco Threat Intelligence	<input checked="" type="checkbox"/> Enabled	142

The enricher task is active, and the enricher runs automatically.

To disable an enricher and prevent it from running, deselect the corresponding **Enabled** checkbox.

The screenshot shows four enricher cards that are inactive:

Enricher Name	Status	Runs this month
PassiveTotal Malware Enricher	<input checked="" type="checkbox"/> Enabled	53
PassiveTotal Passive DNS Enricher	<input type="checkbox"/> Enabled	48
Recorded Future	<input type="checkbox"/> Enabled	121
Splunk Sightings Enricher	<input type="checkbox"/> Enabled	173

The enricher task is inactive, and the task run returned one or more errors.

Run enrichers

Run enrichers, either automatically or manually, to augment existing information with additional context from a range of cyber threat data like IP addresses, domains, email addresses, network streams, hashes, and forums.

Enrich entities

You can enrich entities in the following ways:

- Automatically, or
- Manually.

Enrichment rules and enrichment tasks drive the enrichment process to:

- Poll selected and trustworthy intelligence data sources;
- Retrieve relevant, accurate, and reliable data to augment platform entities with additional bits of information that provide additional context.

Rules

Enrichment rules define what to do with the retrieved enrichment data.

Rules act like filters, and they set the logical constraints defining:

- The platform data sources to augment with enrichment information.
Data sources you can enrich are incoming feeds, other enrichers, and groups.
- Within the selected platform data sources, the entity type(s) to augment with enrichment information.
- The enrichers to use to fetch enrichment data.

Tasks

Enrichment tasks define process execution by setting the following options:

- The data fetching mechanism — for example, an API endpoint exposing the enrichment data service.
- Specific data sources — for example, datasets targeting threat actors like hackers and terrorist groups.
- Data rate limit and monthly execution cap values to control the amount of polled data.
- A source reliability flag for the incoming enrichment data to simplify assessing the quality of the retrieved data.

Enrich automatically

To automatically enrich entities, make sure enricher tasks are active, and the necessary enrichment rules are configured.


Rules give you control over the type of information you want to retrieve or exclude, and what you want to do with it. You can assign one or more enricher sources to specific observable types. You can set multiple filters to cover usage scenarios as needed. You can then examine the returned enrichment observable data, as well as route it to other devices that enforce cyber threat detection or prevention.

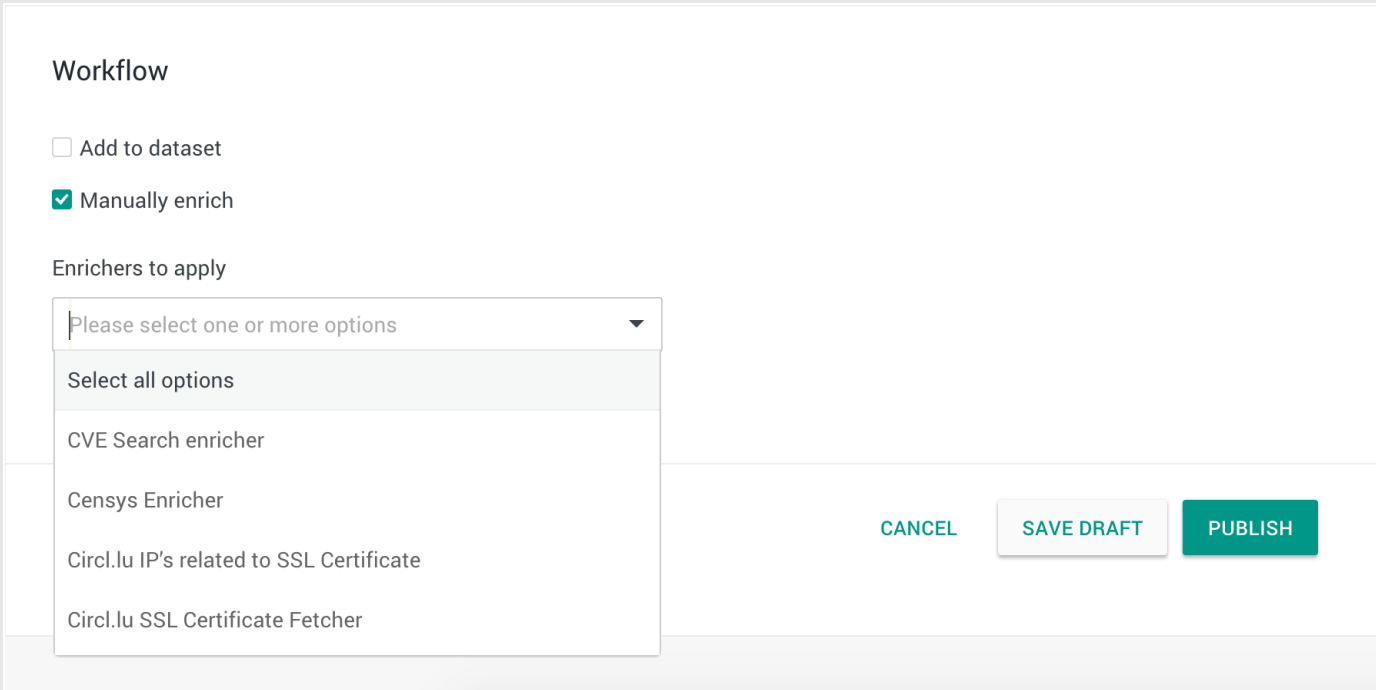
To run the enricher automatically, go to the enricher edit mode, and make sure the **Enabled** checkbox on the edit form is selected.

If it is deselected, check it, and then click **Save**.

Enrich manually

To adjust enrichment behavior to manually apply it to the entities you want to enrich, do the following:

- Open an entity in edit mode.
For example, on the top navigation bar click **Browse > Published** to display an overview of the published entities available in the platform.
- On the row corresponding to the entity you want to manually enrich, click the  icon to display the context menu.
- From the drop-down menu select **Edit**.
- At the bottom of the entity editor page click the **Manually enrich** checkbox.
A new input field with a drop-down menu becomes available.
- From the drop-down menu select one or more enrichers you want to apply to the entity.




- Click **Save draft** to store your changes without publishing the entity, **Publish** to release the new version of the entity including your changes, or **Cancel** to discard the changes.

Alternatively, you can manually enrich an entity by selecting it; for example, from a dataset, from **Browse** or from **Discovery**.

An overlay slides in from the side of the screen to display the entity detail pane.

- On the entity detail pane, click **Observables**.
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.

To manually enrich the entity observables:

- Click the  refresh icon to trigger a task run that polls all the enrichers configured for the entity.

Alternatively:

- From the **Actions** pop-up menu, select **Enrich > Enrich with all**.
- The platform polls all applicable enrichers for the entity, and it enriches all the entity observables with the retrieved data.

Malicious files detected

Ingested: 06.10.2017 9:20 Incoming feed: TAXII Stand Samples

TLP Not Set

OVERVIEW

OBSERVABLES

NEIGHBORHOOD

JSON

VERSIONS

HISTORY

+

<input type="checkbox"/>	Type✓/ Value	Relation	Sighted	Conn.	First seen	Maliciousness	
<input type="checkbox"/>	hash-sha256: e3b0c44298fc1c149afb4c899...	Related +1		2	06.10.2017 9:20		
<input type="checkbox"/>	hash-sha256: d7a8fbb307d7809469ca9abcb...	Related +1		1	06.10.2017 9:20		
<input type="checkbox"/>	file: readme.doc.exe	Related +1		1	06.10.2017 9:20		

Edit

Delete

Add to dataset

Add to graph

Create task

Export

Download original

Enrich

Enrich with all (5)

Censys Enricher

CrowdStrike Enricher

FireEye

Flashpoint AggregINT Enricher


Flashpoint Thresher Enricher

To poll a specific enricher:

- From the **Actions** pop-up menu, select **Enrich**, and then click the specific enricher whose task run you want to trigger.
- The platform polls the specified enricher for the entity, and it enriches all supported entity observables with the retrieved data.

×

Malicious files detected

 Ingested: 06.10.2017 9:20 Incoming feed: TAXII Stand Samples

TLP Not Set

OVERVIEWOBSERVABLESNEIGHBORHOODJSONVERSIONSHISTORY

≡

+

⌵

×

3 selected

Deselect all

Enrich

▼

Add to

▼

⋮

<input checked="" type="checkbox"/>	Type	Value	Relation	Sighted	Conn.	Enrich with all (5)	Refresh
<input checked="" type="checkbox"/>	hash-sha256	e3b0c44298fc1c149afb4c899...	Related +1		2	Censys Enricher	⋮
<input checked="" type="checkbox"/>	hash-sha256	d7a8fbb307d7809469ca9abcb...	Related +1		1	CrowdStrike Enricher	⋮
<input checked="" type="checkbox"/>	file	readme.doc.exe	Related +1		1	FireEye	⋮

Censys Enricher

CrowdStrike Enricher

FireEye

Flashpoint AggregINT Enricher


Flashpoint Thresher Enricher

To enrich only specific observables:

- On the **Observables** tab, select the checkboxes corresponding to the observables you want to enrich.
- From the **Enrich** drop-down menu, select **Enrich with all**.
- The platform polls all applicable enrichers for the entity, and it enriches the selected entity observables with the retrieved data.

×

Malicious files detected

 Ingested: 06.10.2017 9:20 Incoming feed: TAXII Stand Samples

TLP Not Set

OVERVIEWOBSERVABLESNEIGHBORHOODJSONVERSIONSHISTORY

≡

+

⌵

×

2 selected

Deselect all

Enrich

▼

Add to

▼

⋮

<input type="checkbox"/>	Type	Value	Relation	Sighted	Conn.	Enrich with all (5)	Refresh
<input checked="" type="checkbox"/>	hash-sha256	e3b0c44298fc1c149afb4c899...	Related +1		2	Censys Enricher	⋮
<input type="checkbox"/>	hash-sha256	d7a8fbb307d7809469ca9abcb...	Related +1		1	CrowdStrike Enricher	⋮
<input checked="" type="checkbox"/>	file	readme.doc.exe	Related +1		1	FireEye	⋮

Censys Enricher

CrowdStrike Enricher

FireEye

Flashpoint AggregINT Enricher

Flashpoint Thresher Enricher

The available enricher tasks in the drop-down menu are automatically filtered to show only the applicable enrichers for the entity.

Enrichers automatically augment all the entities that accept the enricher's content type as an observable. In other words, the observable types an entity supports define the applicable enrichers an entity can use.

Review enrichment observables

To view enrichment information on the entity detail pane, do the following:


- Select an entity; for example, from a dataset, from **Browse** or from **Discovery**.
An overlay slides in from the side of the screen to display the entity detail pane.
- On the entity detail pane, click **Observables**.
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.











The screenshot displays the 'Observables' tab within an entity detail pane. The header bar is teal and contains a fingerprint icon, the text 'Ingested: Yesterday at 23:59 Incoming feed: tor', and a 'TLP White' button. Below the header, a navigation bar shows tabs: OVERVIEW, OBSERVABLES (selected), NEIGHBORHOOD, JSON, VERSIONS, and HISTORY. The main content area features a table of enrichment observables. A tooltip 'Add observable' points to a plus icon in the top left of the table. The table has columns: Type / Value, Relation, Sighted, Conn., First seen, and Maliciousness. The first row shows a domain observable 'torstatus.blutmagie.de' with a 'Description' relation and a count of '+1'. Subsequent rows show various hash and email observables with 'OpenResolve' or 'Recorded Future' relations and counts of '+4' or '+1'. The 'Maliciousness' column uses red and grey dots to indicate risk levels. At the bottom, a pagination bar shows '1 - 10 of 123' and navigation arrows.

Type / Value	Relation	Sighted	Conn.	First seen	Maliciousness
domain: torstatus.blutmagie.de	Description + 1		96264	02.08.2017 14:52	●●●
ipv4: 194.63.141.179	OpenResolve + 4			02.08.2017 14:53	●●●
email: registry@regfish.com	Threatcrowd AP...			02.08.2017 14:53	●●●
hash-md5: e63989f8df535a14cee4a06484c...	Recorded Future			02.08.2017 14:53	●●●
hash-sha256: c27c743c64713fd2a2b36b32...	Recorded Future			02.08.2017 14:53	●●●
hash-sha512: 5a9df660747757935abfcdf4...	Recorded Future			02.08.2017 14:53	●●●
email: bitwisser@googlemail.com	Recorded Future			02.08.2017 14:53	●●●
hash-sha1: 4e0c92f8e8dca4d9295c7587e7...	Recorded Future			02.08.2017 14:53	●●●
hash-md5: b7532bac9c0d65b3a4a724a603...	Recorded Future			02.08.2017 14:53	●●●
hash-sha256: 6ae3032f45266fd48589bfa8...	Recorded Future			02.08.2017 14:53	●●●

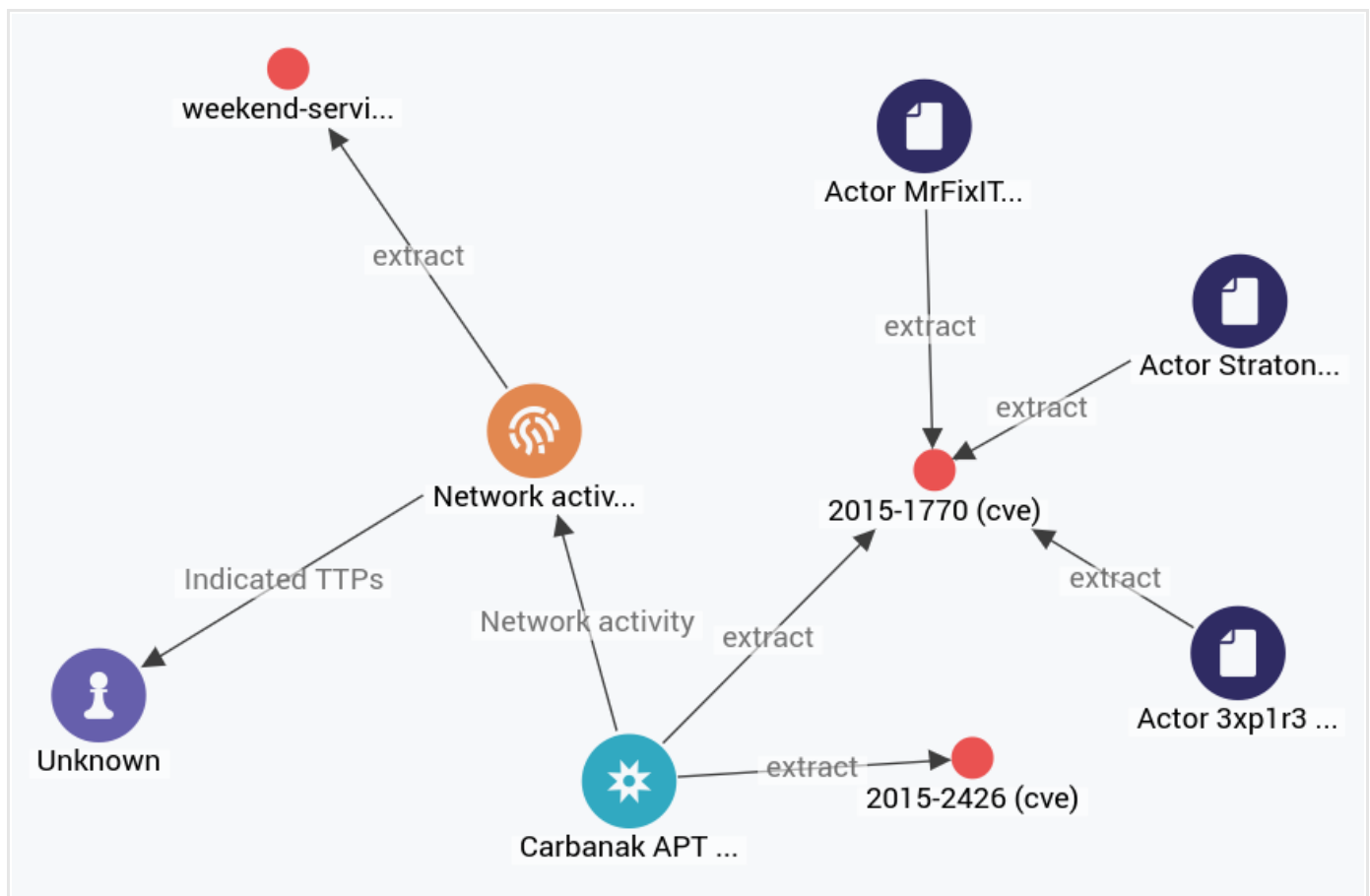
Review enrichment observables on the graph

To view enrichment data and their connections with other entities and observables on the graph, do the following:

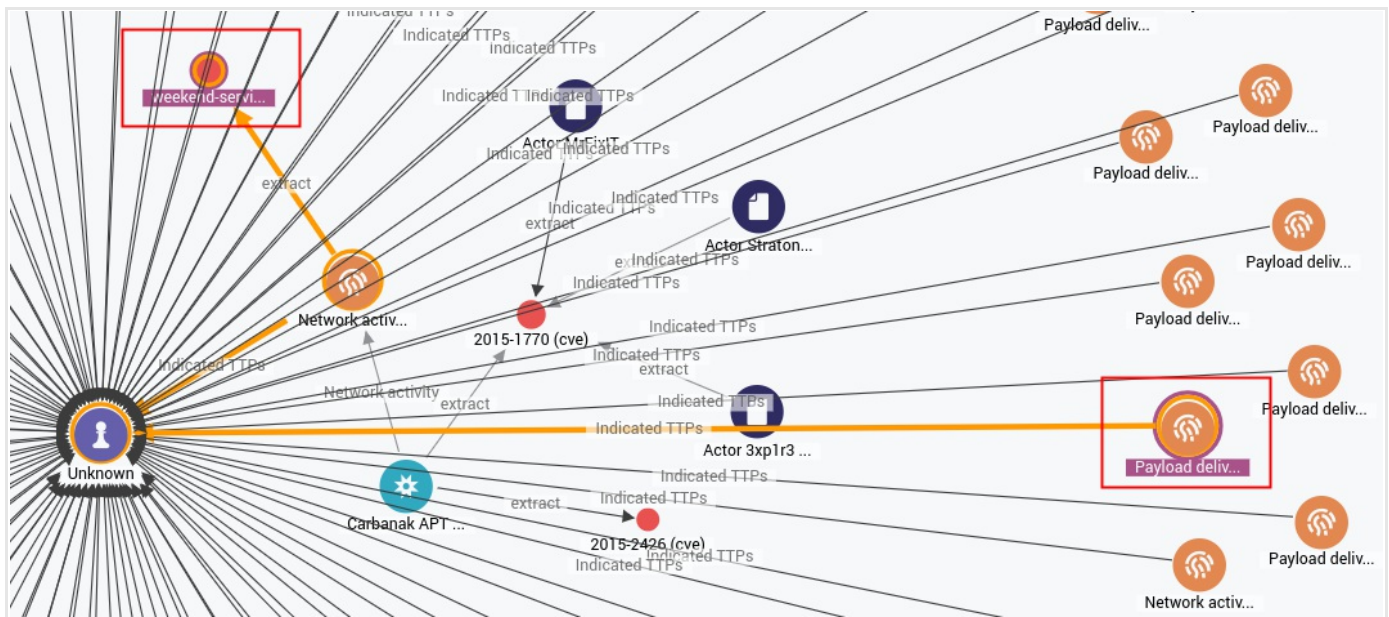
- On the row corresponding to the observable you want to load on the graph, click the  icon, and then select **Add to graph**.

ENTITIES		OBSERVABLES	UPLOADS	DATASETS	SAVED GRAPHS						
		Filter...									
<input type="checkbox"/>	Type	Value			Sightings	Maliciousness	Connections	Source	Timestamp		
<input type="checkbox"/>	ipV4	217.93.38.31					1	tor	29.08.2017 9:23		
<input type="checkbox"/>	domain	pd95d261f.dip0.t-ipconnect.de					1	tor			
<input type="checkbox"/>	ipV4	93.238.180.183					1	Gecko Indicator...			
<input type="checkbox"/>	hash-md5	961b27607e5c2f7915558e1cee520bb4					7	Gecko CS indie...			
<input type="checkbox"/>	uri	https://www.threatq.com/p42/search/default?m=7413					1	Gecko Phishme...			

- To load the parent entity whose detail pane you are viewing, instead of its observables, from the pop-up **Actions** menu at the bottom of the pane select **Add to graph**.
- Click the graph thumbnail on the lower side of the screen to expand it.
- On the graph, right-click the entity you want to inspect, and from the context menu select **Load entities > All**, **Load observables > All** or **Load entities by extract > All**.

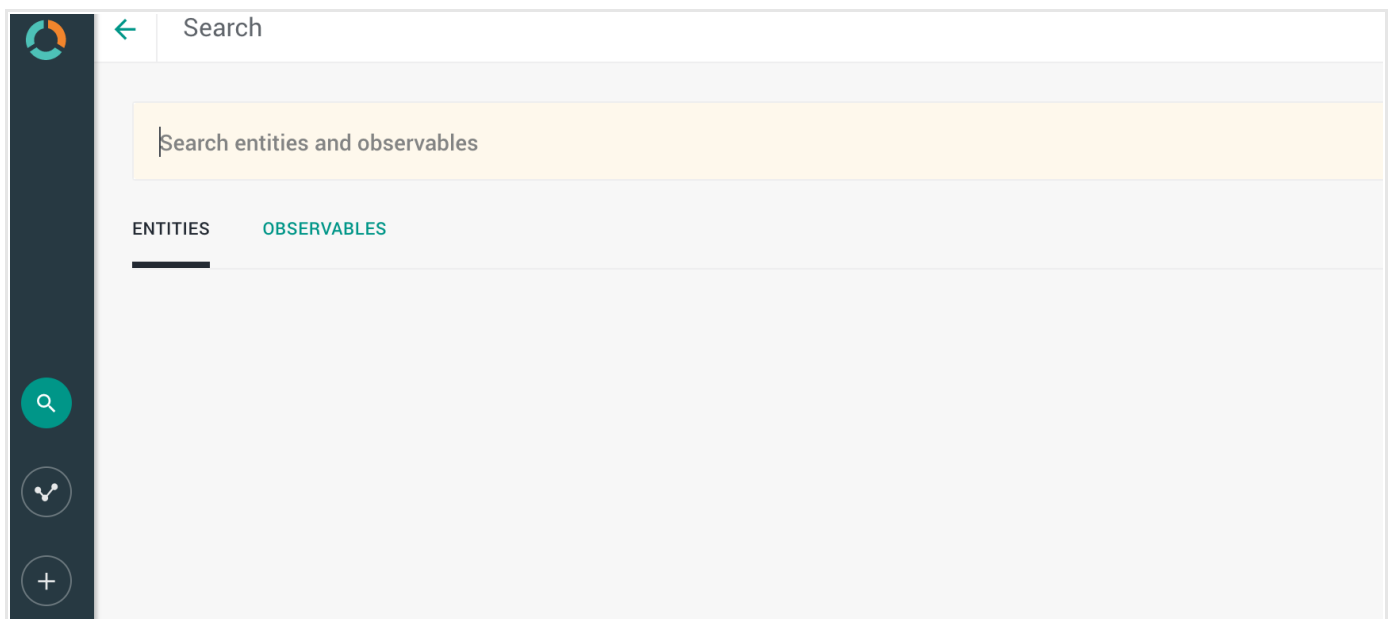


- Right-click an extract or an entity for further inspection and from the context menu select **Load entities > All**, **Load observables > All** or **Load entities by extract > All**.



Search for enrichment observables

You can use the search box to look for enrichment observables. You can find the search box on the sidebar:



Quick search: Hover over the magnifier and enter search queries. Click the search icon to run the search.

Specific search: click the magnifier and enter search terms and search queries. Then click **ENTER** or click the search icon to run the search.

Searches you run through this search box are executed platform-wide.

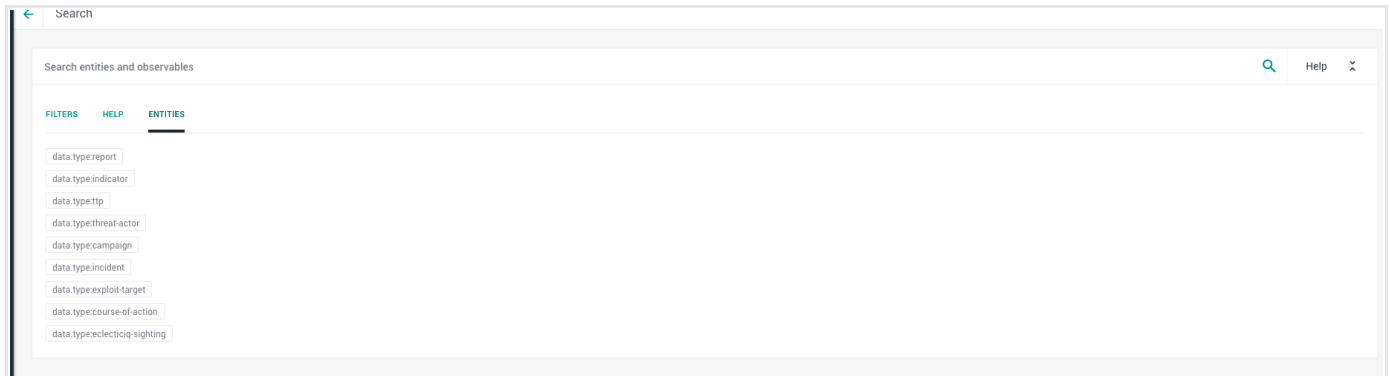


The search functionality uses **Elasticsearch query syntax**

(<https://www.elastic.co/guide/en/elasticsearch/reference/current/full-text-queries.html>).

To access a cheatsheet with search examples using entity types, filters, and for help with the search syntax, click **Help** to display thematic drop-down lists with common search queries:

- **Filters:** examples of quick search filters.
- **Help:** examples of regex, Boolean, wildcards, and tag search usage.
- **Entities:** examples of searchable entity types.

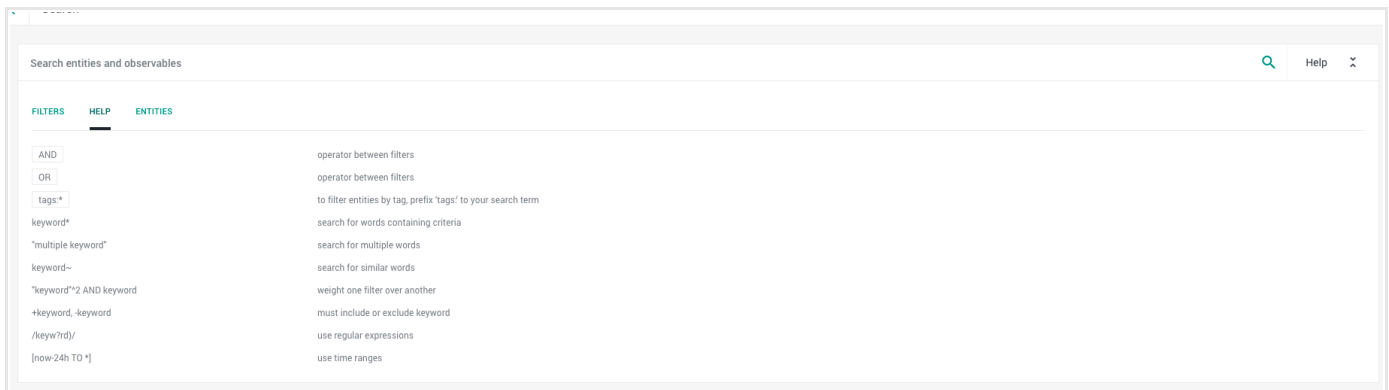


About wildcards

- Currently supported **wildcards** (<https://www.elastic.co/guide/en/elasticsearch/reference/5.6/query-dsl-wildcard-query.html>): * and ?
- Escape character: \
- Avoid starting a wildcard query term with * or ?, as it may result in an extremely slow-running query.

About complex searches

Besides full text search, you can use Boolean operators, wildcards, regex, and you can combine these filtering options to create more refined searches.



Use operators to combine multiple quick filters and create a more complex search query.

Search example:

```
enrichment_extracts.meta.blacklisted:true AND enrichment_extracts.kind:hash-md5
```

Search result example:

```
{
  "instance_meta": {
    "classification": "bad",
    "confidence": "low",
    "blacklisted": true
  },

  "id": 123456,
  "value": "7cd7ekhc5f5742fdccf655767b15h7g78",

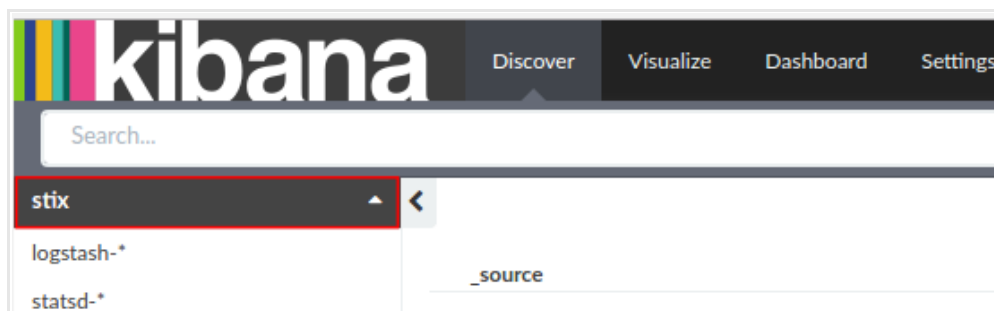
  "meta": {
    "classification": "bad",
    "confidence": "low",
    "blacklisted": true
  },

  "kind": "hash-md5",
  "instance_id": 123456
}
```

Field	Description	Example
<i>enrichment_extracts.id</i>	string — The alphanumeric ID string that uniquely identifies the enrichment observable.	01h12x45-01q2-1234-od01-123456h78h90
<i>enrichment_extracts.kind</i>	string — The enrichment observable data type.	domain
<i>enrichment_extracts.meta.blacklisted</i>	Boolean — An observable is blacklisted when it is included in the results returned by an <i>ignore</i> extraction rule. Allowed values: <code>true</code> , <code>false</code> .	true
<i>enrichment_extracts.meta.classification</i>	string — This value is defined in Rules by selecting one of the available options under Action and Confidence . Allowed classification metadata values are <code>good</code> , <code>bad</code> , and <code>unknown</code> .	good
<i>enrichment_extracts.meta.confidence</i>	string — This value is defined in Rules by selecting the appropriate option under Action and Confidence . The selected action must be Mark as malicious for the Confidence drop-down list to become available. Allowed confidence metadata values are <code>low</code> , <code>medium</code> , and <code>high</code> .	high
<i>enrichment_extracts.value</i>	string — The actual value of the enrichment observable, based on the enrichment observable data type.	doom.dismay.biz

For reference, you can look up a complete list of all available search query fields in Kibana:

- To access Kibana, in the web browser address bar enter a URL with the following format:
[https://\\${platform_host}/private/kibana/app/kibana#](https://${platform_host}/private/kibana/app/kibana#)
 Keep the trailing #
 Example: [https://\\${platform_host}.com/private/kibana/app/kibana#](https://${platform_host}.com/private/kibana/app/kibana#)
- Select the **stix** index field:



- On the main menu bar, select **Settings**:

A screenshot of the Kibana Settings page for the 'stix' index pattern. The top navigation bar includes 'Discover', 'Visualize', 'Dashboard', and 'Settings' (which is highlighted). Below this, a sub-menu shows 'Indices', 'Advanced', 'Objects', and 'About', with 'Indices' selected. On the left sidebar, under 'Index Patterns', 'stix' is selected. The main content area shows the 'stix' index pattern with a star, refresh, and delete icon. A description states: 'This page lists every field in the stix index and the field's associated core type as recorded by Elasticsearch. While this list allows you to view the core type of each field, changing field types must be done using Elasticsearch's Mapping API'. Below this, there are two tabs: 'Fields (428)' and 'Scripted fields (0)'. A table lists the fields with columns: name, type, format, analyzed, indexed, and controls. The table contains 15 rows of field data.

name	type	format	analyzed	indexed	controls
data.kill_chain_phases.kill_chain_name	string		✓	✓	
data.observable.object.related_objects.related_objects.relationship	string		✓	✓	
data.observable.composition.composition.composition.type	string		✓	✓	
data.producer.contributing_sources.type	string		✓	✓	
data.observable.object.related_objects.related_objects.properties_xml_type	string		✓	✓	
exposure.affected_overrides.state	boolean			✓	
data.test_mechanisms.rules.value	string		✓	✓	
data.indicated_ttps.idref	string		✓	✓	
data.handling.marking_structures.marking_structure_type	string		✓	✓	
exposure.sighted	boolean			✓	
exposure.prevent_ok	boolean			✓	
destinations	string			✓	
tags	string		✓	✓	

Censys enricher

The Censys enricher returns a wealth of information about IP addresses, from a network ASN to their geographic location, so that you can explore relationships between events, actors, and targets.

This article describes how to configure the Censys enricher parameters.

To configure the general options for the Censys enricher, see [Configure enrichers](#).

Censys	enricher
Enricher name	Censys
API endpoint	https://censys.io/api/v1/search/ipv4
Input	asn, city, company, country, country_code, geo-lat, geo-long, hash-md5, hash-sha1, hash-sha256, ipv4, postcode
Output	Enriches supported observable types by providing additional context such as geolocation, country and city information, as well as ASN (https://en.wikipedia.org/wiki/autonomous_system_(internet)) details.
Description	Returns relevant contextual information about the submitted observable types to augment their intelligence value with geographic and geolocation details, hashes, and ASN (https://en.wikipedia.org/wiki/autonomous_system_(internet)) details. It makes it easier to discover relationships between events, actors, and targets.

Configure the Censys enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Censys enricher.

Supported observable types:

- *asn*
- *city*
- *company*
- *country*
- *country_code*
- *geo-lat*
- *geo-long*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *ipv4*
- *postcode*

Under **Parameters**, define the specific configuration options for the Censys enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
The API URL for this enricher exposes the **Censys Search API** (<https://censys.io/api/v1/docs/search>).
The *ipv4* URL parameter allows sending requests to the IP address search index.
- **API ID: create an account** (<https://censys.io/register>) to receive the login credentials you need to authenticate and access the API service.
Enter here your API user ID.
- **API secret:** enter the secret key associated with your API user profile, so that you can log in and consume the API service.

- **Observable queries:** from the drop-down menu select the observable type and the corresponding observable value the rule should look for.
 - In the first input field, from the drop-down menu select the *observable type* the rule should look for. Supported observable types:
 - *asn*
 - *city*
 - *company*
 - *country*
 - *country_code*
 - *geo-lat*
 - *geo-long*
 - *hash-md5*
 - *hash-sha1*
 - *hash-sha256*
 - *ipv4*
 - *postcode*
 - In the second input field, specify the *observable value* associated with the observable type that the rule should look for.

You can use free text, wildcards, **Elasticsearch query syntax**

(<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html>), as well as the *{kind}* and *{value}* placeholders to reference an observable type and value, respectively.

When the query executes, the placeholders take the values from the input observable key (*{kind}*) and value (*{value}*) pairs, respectively.

Example:

The **@{value}* query searches for observable values matching the input observable values it is fed at runtime.

Censys allows using **specific data fields** (<https://censys.io/overview>) to search for data related to IP hosts. You can combine these data definitions with the *{kind}* and *{value}* placeholders.

- Click **+ Add** or **+ More** to add a new filtering option. For example, to include in the search additional key/value pairs like IP addresses, hashes, or domains.
- Click **Save** to store your changes, or **Cancel** to discard them.

Query the Censys API

Observable queries accept key/value pairs:

- In the first input field you define the type of observable you want to search for.
- In the second input field you specify the data pattern to use in the search to look for the desired values.

The Censys **Search API** (<https://censys.io/api/v1/docs/search>) allows simple searches for words or phrases, as well as complex searches using query syntax and query parameters:

- For the search syntax, refer to **Censys Overview** (<https://censys.io/overview>), under **Search Syntax**.

- For the query parameters you can pass, refer to **Censys Overview** (<https://censys.io/overview>), under **Data Definitions > IPv4 Hosts**.

When you chain multiple parameters in the same query without specifying any Boolean operator, the default Boolean operator between parameters is *or*.

The table overview shows query examples you can enter under **Observable queries** to search for specific observable types and values.

First query field	Second query field
<i>asn</i>	<i>autonomous_system.asn</i> : {value}
<i>city</i>	<i>location.city</i> : "{value}"
<i>company</i>	<i>metadata.manufacturer</i> : "{value}"
<i>country</i>	<i>location.country</i> : "{value}"
<i>country-code</i>	<i>location.country_code</i> : "{value}"
<i>geo-lat</i>	<i>location.latitude</i> : {value}
<i>geo-long</i>	<i>location.longitude</i> : {value}
<i>ipv4</i>	<i>ip</i> : {value}
<i>postcode</i>	<i>location.postal_code</i> : "{value}"
<i>hash-md5</i>	<i>25.smtp.starttls.tls.certificate.parsed.fingerprint_md5</i> : "{value}" or <i>25.smtp.starttls.tls.chain.parsed.fingerprint_md5</i> : "{value}" or <i>80.http.get.headers.content_md5</i> : "{value}" or <i>80.http.open_proxy.connect.headers.content_md5</i> : "{value}" or <i>80.http.open_proxy.get.headers.content_md5</i> : "{value}" or <i>143.imap.starttls.tls.certificate.parsed.fingerprint_md5</i> : "{value}" or <i>143.imap.starttls.tls.chain.parsed.fingerprint_md5</i> : "{value}" or <i>443.https.tls.certificate.parsed.fingerprint_md5</i> : "{value}" or <i>443.https.tls.chain.parsed.fingerprint_md5</i> : "{value}" or <i>110.pop3.starttls.tls.certificate.parsed.fingerprint_md5</i> : "{value}" or <i>110.pop3.starttls.tls.chain.parsed.fingerprint_md5</i> : "{value}" or <i>993.imaps.tls.tls.certificate.parsed.fingerprint_md5</i> : "{value}" or <i>993.imaps.tls.tls.chain.parsed.fingerprint_md5</i> : "{value}" or <i>995.pop3s.tls.tls.certificate.parsed.fingerprint_md5</i> : "{value}" or <i>995.pop3s.tls.tls.chain.parsed.fingerprint_md5</i> : "{value}"
<i>hash-sha1</i>	<i>25.smtp.starttls.tls.certificate.parsed.fingerprint_sha1</i> : "{value}" or <i>25.smtp.starttls.tls.chain.parsed.fingerprint_sha1</i> : "{value}" or <i>143.imap.starttls.tls.certificate.parsed.fingerprint_sha1</i> : "{value}" or <i>143.imap.starttls.tls.chain.parsed.fingerprint_sha1</i> : "{value}" or <i>443.https.tls.certificate.parsed.fingerprint_sha1</i> : "{value}" or <i>443.https.tls.chain.parsed.fingerprint_sha1</i> : "{value}" or <i>110.pop3.starttls.tls.certificate.parsed.fingerprint_sha1</i> : "{value}" or <i>110.pop3.starttls.tls.chain.parsed.fingerprint_sha1</i> : "{value}" or <i>993.imaps.tls.tls.certificate.parsed.fingerprint_sha1</i> : "{value}" or <i>993.imaps.tls.tls.chain.parsed.fingerprint_sha1</i> : "{value}" or <i>995.pop3s.tls.tls.certificate.parsed.fingerprint_sha1</i> : "{value}" or <i>995.pop3s.tls.tls.chain.parsed.fingerprint_sha1</i> : "{value}"

First query field	Second query field
<i>hash-sha256</i>	25.smtp.starttls.tls.certificate.parsed.fingerprint_sha256: "{value}" or 25.smtp.starttls.tls.chain.parsed.fingerprint_sha256: "{value}" or 80.http.get.body_sha256: "{value}" or 80.http.open_proxy.get.body_sha256: "{value}" or 143.imap.starttls.tls.certificate.parsed.fingerprint_sha256: "{value}" or 143.imap.starttls.tls.chain.parsed.fingerprint_sha256: "{value}" or 443.https.tls.certificate.parsed.fingerprint_sha256: "{value}" or 443.https.tls.chain.parsed.fingerprint_sha256: "{value}" or 110.pop3.starttls.tls.certificate.parsed.fingerprint_sha256: "{value}" or 110.pop3.starttls.tls.chain.parsed.fingerprint_sha256: "{value}" or 993.imaps.tls.tls.certificate.parsed.fingerprint_sha256: "{value}" or 993.imaps.tls.tls.chain.parsed.fingerprint_sha256: "{value}" or 995.pop3s.tls.tls.certificate.parsed.fingerprint_sha256: "{value}" or 995.pop3s.tls.tls.chain.parsed.fingerprint_sha256: "{value}"



By default, the Censys enricher timeout value is set to *1 minute*.

CIRCL IPs related to SSL certificate enricher

The CIRCL IPs related to SSL certificate enricher uses the CIRCL API to poll the CIRCL Passive SSL database and to obtain all IP addresses associated with the input SSL SHA-1 hash fingerprints.

This article describes how to configure the CIRCL IPs related to SSL certificate enricher parameters. To configure the general options for the CIRCL IPs related to SSL certificate enricher, see [Configure enrichers](#).

CIRCL IPs related to SSL certificate	enricher
Enricher name	CIRCL IPs related to SSL certificate
API endpoint	<code>https://www.circl.lu/v2pssl/cquery/{}</code>
Input	hash-sha1
Output	Enriches SSL fingerprint hash observables with all associated IP addresses.
Description	The enricher contacts the Computer Incident Response Center Luxembourg (CIRCL) API and the CIRCL Passive SSL database to retrieve all observed IP addresses associated with the input SSL fingerprint <i>hash-sha1</i> observables. The CIRCL Passive SSL database holds historical data matching SSL certificates and IP addresses.

Configure the CIRCL IPs related to SSL certificate enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the CIRCL IPs related to SSL certificate enricher.
Supported observable types:
 - *hash-sha1*

Under **Parameters**, define the specific configuration options for the CIRCL IPs related to SSL certificate enricher:

- **API user name:** sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key: contact CIRCL** (<https://www.circl.lu/contact/>) to receive an API key, and then enter it in the corresponding input field.
- Click **Save** to store your changes, or **Cancel** to discard them.

i By default, the CIRCL IPs related to SSL certificate enricher timeout value is set to *1 minute*.

CIRCL SSL Certificate Fetcher enricher

The CIRCL SSL Certificate Fetcher enricher uses the CIRCL API to poll the CIRCL Passive SSL database and to obtain parsed SSL certificates and all domain names associated with the input SSL SHA-1 hash fingerprints.

This article describes how to configure the CIRCL SSL Certificate Fetcher enricher parameters. To configure the general options for the CIRCL SSL Certificate Fetcher enricher, see [Configure enrichers](#).

CIRCL SSL Certificate Fetcher	enricher
Enricher name	CIRCL SSL Certificate Fetcher
API endpoint	<code>https://www.circl.lu/v2pssl/cfetch/{}</code>
Input	hash-sha1
Output	Enriches SSL fingerprint hash observables with the parsed certificate and all associated domain names.
Description	The enricher contacts the Computer Incident Response Center Luxembourg (CIRCL) API and the CIRCL Passive SSL database to retrieve the parsed certificate and all observed domain names associated with the input SSL fingerprint <i>hash-sha1</i> observables. The CIRCL Passive SSL database holds historical data matching SSL certificates fingerprints with the corresponding parsed certificates and the related domain names, that is, the certificate <code>subject CN</code> and <code>subjectAltName/Subject Alternative Name (SAN)</code> , if available.

Configure the CIRCL SSL Certificate Fetcher enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the CIRCL SSL Certificate Fetcher enricher.
Supported observable types:

- *hash-sha1*

Under **Parameters**, define the specific configuration options for the CIRCL SSL Certificate Fetcher enricher:

- **API user name:** sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key: contact CIRCL** (<https://www.circl.lu/contact/>) to receive an API key, and then enter it in the corresponding input field.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the CIRCL SSL Certificate Fetcher enricher timeout value is set to *1 minute*.

Cisco Threat Grid enricher

The Cisco Threat Grid enricher provides information on a range of cyber threat data like IP addresses, domains, registry keys, network streams, and hash files.

This article describes how to configure the Cisco Threat Grid enricher parameters.

To configure the general options for the Cisco Threat Grid enricher, see [Configure enrichers](#).

Cisco Threat Grid	enricher
Enricher name	Cisco Threat Grid
API endpoint	<code>https://panacea.threatgrid.com/api/v3/</code>
Input	domain, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri, winregistry
Output	Enriches supported observable types, as well as all found observables based on the enricher configuration, with information such as IP addresses, domains, host names, hashes, and Windows registry keys.
Description	Polls data from the Cisco Threat Grid API. It provides information on a range of cyber threat data like IP addresses, domains, registry keys, network streams, and hash files.

Configure the Cisco Threat Grid enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Cisco Threat Grid enricher.

Supported observable types:

- *domain*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *hash-sha512*
- *host*
- *ipv4*
- *ipv6*
- *uri*
- *winregistry*

Under **Parameters**, define the specific configuration options for the Cisco Threat Grid enricher:

- **API URL**: the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
- **API key**: contact Cisco to receive an API key, and then enter it in the corresponding input field.
- **Organization only**: select this checkbox to enable the enricher to check and display only submitted samples created by the organization the current user belongs to. That is, the organization needs to be the author of the submitted samples. When selected, this field is validated against the API key value granting access to the service.
- **Max low confidence threat score**: you can set an *upper threshold* to automatically flag enriched observables with a *low confidence* value.
After completing the sample analysis, enriched observables with a *lower* threat score than the specified value are flagged as **Malicious - Low confidence**.
 - Enter an integer value between 0 and 100.
 - Default value: 85.
- **Min high confidence threat score**: you can set a *bottom threshold* to automatically flag enriched observables with a *high confidence* value.
After completing the sample analysis, enriched observables with a *higher* threat score than the specified value are flagged as **Malicious - High confidence**.
 - Enter an integer value between 0 and 100.
 - Default value: 95.
- Enriched observables with a threat score falling in the range defined by **Max low confidence threat score** (range lower limit) and **Min high confidence threat score** (range upper limit) are flagged as **Malicious - Medium confidence**.
- Click **Save** to store your changes, or **Cancel** to discard them.



Automatic flagging with high/low confidence maliciousness through Cisco Threat Grid supports only the following observable types:

- *hash-sha1*
- *hash-sha256*
- *hash-md5*
- *uri*

By default, the Cisco Threat Grid enricher timeout value is set to 5 *minutes*.

Crowdstrike Falcon Intelligence Indicator enricher

The Crowdstrike Falcon Intelligence Indicator enricher returns observables extracted from indicators to provide additional context to existing platform intelligence.

This article describes how to configure the Crowdstrike Falcon Intelligence Indicator enricher parameters. To configure the general options for the Crowdstrike Falcon Intelligence Indicator enricher, see [Configure enrichers](#).

Crowdstrike Falcon Intelligence Indicator	enricher
Enricher name	Crowdstrike Falcon Intelligence Indicator
API endpoint	<code>https://intelapi.crowdstrike.com/indicator/v1/search/{}</code>
Input	domain, email, email-subject, file, hash-md5, hash-sha1, hash-sha256, ipv4, ipv6, mutex, name, persona, port, uri
Output	Enriches supported observable types with information extracted from indicators.
Description	Enriches platform entities and observables with additional context such as IP addresses, domain names, email addresses, hashes, and more.

Configure the Crowdstrike Falcon Intelligence Indicator enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the CrowdStrike Falcon Intelligence Indicator enricher.

Supported observable types:

- *domain*
- *email*
- *email-subject*
- *file*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *ipv4*
- *ipv6*
- *mutex*
- *name*
- *persona*
- *port*
- *uri*

Under **Parameters**, define the specific configuration options for the CrowdStrike Falcon Intelligence Indicator enricher:

- **API ID:** contact CrowdStrike to receive an API ID, and then enter it in the corresponding input field.
You need a valid API ID and a corresponding API key as authentication credentials to access the CrowdStrike Falcon Intel API and to consume it.
- **API key:** contact CrowdStrike to receive an API key, and then enter it in the corresponding input field.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the CrowdStrike Falcon Intelligence Indicator enricher timeout value is set to *1 minute*.

CVE Search enricher

The CVE Search enricher contacts CIRCL cve-search API to poll information about common software and hardware vulnerabilities, along with the corresponding exposures.

This article describes how to configure the CVE Search enricher parameters.
To configure the general options for the CVE Search enricher, see [Configure enrichers](#).

CVE Search	enricher
Enricher name	CVE Search
API endpoint	<code>https://cve.circl.lu/api/cve/</code>
Input	cve
Output	Enriches supported observable types with CVE details.
Description	Enriches supported observable types with information about common software and hardware vulnerabilities, along with the corresponding exposures. The enricher contacts the Computer Incident Response Center Luxembourg (CIRCL) cve-search API to retrieve all the available details associated with the input CVE IDs. CVE information about common software and hardware vulnerabilities, along with the corresponding exposures, is stored in the platform as enrichment observables.

Configure the CVE Search enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the CVE Search enricher.
Supported observable types:
 - cve

Under **Parameters**, define the specific configuration options for the CVE Search enricher:

- **CVE API URL :** the URL pointing to the API endpoint exposing the service that grants access to the enricher data. Contact the intel provider to subscribe to the service and to obtain this information.
For this enricher it corresponds to `https://cve.circl.lu/api/cve/`.
- Click **Save** to store your changes, or **Cancel** to discard them.

i By default, the CVE Search enricher timeout value is set to *1 minute*.

DomainTools Hosted Domains enricher

The Domaintools Hosted Domains enricher returns all domain names related to the specified input IP addresses.

This article describes how to configure the DomainTools Hosted Domains enricher parameters. To configure the general options for the DomainTools Hosted Domains enricher, see [Configure enrichers](#).

DomainTools Hosted Domains	enricher
Enricher name	DomainTools Hosted Domains
API endpoint	<code>http://api.domaintools.com/v1/{}/host-domains</code>
Input	ipv4
Output	Enriches supported observable types with domain and host name information.
Description	Enriches IPv4 observables by returning all the domain names hosted on, and therefore related to, the input IP addresses.

Configure the DomainTools Hosted Domains enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the DomainTools Hosted Domains enricher.
Supported observable types:

- *ipv4*

Under **Parameters**, define the specific configuration options for the DomainTools Hosted Domains enricher:

- **Username:** sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key:** contact DomainTools to receive an API key, and then enter it in the corresponding input field.
- Click **Save** to store your changes, or **Cancel** to discard them.

i By default, the DomainTools Hosted Domains enricher timeout value is set to *1 minute*.

DomainTools Malicious Server Domains enricher

The DomainTools Malicious Server Domains enricher returns malicious domain names related to the same primary and/or secondary name servers, along with their risk scores to automatically flag server domains with an appropriate maliciousness confidence level.

This article describes how to configure the DomainTools Malicious Server Domains enricher parameters. To configure the general options for the DomainTools Malicious Server Domains enricher, see [Configure enrichers](#).

DomainTools Malicious Server Domains	enricher
Enricher name	DomainTools Malicious Server Domains
API endpoint	<code>http://api.domaintools.com/v1/{}/name-server-domains/</code>
Input	domain, host
Output	Enriches supported observable types with malicious domain names that are served from the same name server.
Description	Enriches domain and host observable types with a list of malicious domain names related to the same primary or secondary name server. It includes configurable thresholds to assign maliciousness confidence levels to the processed domains and hosts, and to ignore non-malicious domains/hosts.

Configure the DomainTools Malicious Server Domains enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the DomainTools Malicious Server Domains enricher.

Supported observable types:

- *domain*
- *host*

Under **Parameters**, define the specific configuration options for the DomainTools Malicious Server Domains enricher:

- **Username:** sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key:** contact DomainTools to receive an API key, and then enter it in the corresponding input field.

- **Low maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - Low confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *low maliciousness threshold* and lower than the medium and high maliciousness thresholds are flagged as **Malicious - Low confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 10.
- **Medium maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - Medium confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *medium maliciousness threshold* and lower than the high maliciousness threshold are flagged as **Malicious - Medium confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 40.
- **High maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - High confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *high maliciousness threshold* are flagged as **Malicious - High confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 80.
- **Ignore non-malicious domains:** select this checkbox to to exclude from ingestion any domains and hosts whose reputation/risk score value is lower than the specified *low maliciousness threshold*.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the DomainTools Malicious Server Domains enricher timeout value is set to 1 minute.

DomainTools Reputation enricher

The DomainTools Reputation enricher returns risk scores to assess the reputation of the specified input domain and host names.

This article describes how to configure the DomainTools Reputation enricher parameters. To configure the general options for the DomainTools Reputation enricher, see [Configure enrichers](#).

DomainTools Reputation	enricher
Enricher name	DomainTools Reputation
API endpoint	<code>http://api.domaintools.com/v1/reputation</code>
Input	domain, host
Output	Enriches supported observable types with reputation information.
Description	Enriches domain and host name observables with reputation/risk score information to assess maliciousness confidence levels, based on user-defined threshold values.

Configure the DomainTools Reputation enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the DomainTools Reputation enricher.
Supported observable types:

- *domain*
- *host*

Under **Parameters**, define the specific configuration options for the DomainTools Reputation enricher:

- **Username:** sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key:** contact DomainTools to receive an API key, and then enter it in the corresponding input field.
- **Low maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - Low confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *low maliciousness threshold* and lower than the medium and high maliciousness thresholds are flagged as **Malicious - Low confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 10.

- **Medium maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - Medium confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *medium maliciousness threshold* and lower than the high maliciousness threshold are flagged as **Malicious - Medium confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 40.
- **High maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - High confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *high maliciousness threshold* are flagged as **Malicious - High confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 80.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the DomainTools Reputation enricher timeout value is set to 1 minute.

DomainTools Parsed Whois enricher

The DomainTools Parsed Whois enricher returns malicious domain names related to the same primary and/or secondary name servers.

This article describes how to configure the DomainTools Parsed Whois enricher parameters.

To configure the general options for the DomainTools Parsed Whois enricher, see [Configure enrichers](#).

DomainTools Parsed Whois	enricher
Enricher name	DomainTools Parsed Whois
API endpoint	<code>http://api.domaintools.com/v1/{}/whois/parsed</code>
Input	domain, host, ipv4
Output	Enriches supported observable types with structured Whois information.
Description	Enriches domains, hosts, and IP addresses with Whois information. The JSON output includes the most recent Whois record for the requested domain or IP range, as well as parsed, structured data such as registrant, registrar, contacts, and so on. It helps searching for, indexing, and cross-referencing data in a set of Whois records.

Configure the DomainTools Parsed Whois enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the DomainTools Parsed Whois enricher.

Supported observable types:

- *domain*
- *host*
- *ipv4*

Under **Parameters**, define the specific configuration options for the DomainTools Parsed Whois enricher:

- **Username:** sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key:** contact DomainTools to receive an API key, and then enter it in the corresponding input field.
- Click **Save** to store your changes, or **Cancel** to discard them.

i By default, the DomainTools Parsed Whois enricher timeout value is set to *1 minute*.

DomainTools Suspicious Domains enricher

The DomainTools Suspicious Domains enricher returns suspicious and potentially malicious domains related to the input IP addresses, along with their risk scores to automatically flag domains with an appropriate maliciousness confidence level.

This article describes how to configure the DomainTools Suspicious Domains enricher parameters. To configure the general options for the DomainTools Suspicious Domains enricher, see [Configure enrichers](#).

DomainTools Suspicious Domains	enricher
Enricher name	DomainTools Suspicious Domains
API endpoint	<code>https://api.domaintools.com/v1/{}/host-domains</code>
Input	ipv4
Output	Enriches supported observable types with suspicious domain and host name information.
Description	Enriches IPv4 observables with suspicious domains related to the input IP addresses. It includes configurable thresholds to assign maliciousness confidence levels to the processed IP addresses, and to ignore non-malicious IPs.

Configure the DomainTools Suspicious Domains enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types**: select one or more observable types you want to enrich with data retrieved through the DomainTools Suspicious Domains enricher.
Supported observable types:

- *ipv4*

Under **Parameters**, define the specific configuration options for the DomainTools Suspicious Domains enricher:

- **Username**: sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key**: contact DomainTools to receive an API key, and then enter it in the corresponding input field.
- **Low maliciousness threshold**: domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - Low confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *low maliciousness threshold* and lower than the medium and high maliciousness thresholds are flagged as **Malicious - Low confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 10.

- **Medium maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - Medium confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *medium maliciousness threshold* and lower than the high maliciousness threshold are flagged as **Malicious - Medium confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 40.
- **High maliciousness threshold:** domain and host names with a higher DomainTools risk score than the value defined here are flagged as **Malicious - High confidence**.
After completing the analysis, enriched domain and host names with a *higher* risk score than the *high maliciousness threshold* are flagged as **Malicious - High confidence**.
 - Enter a value between 0 and 99.99.
 - Default value: 80.
- **Ignore non-malicious domains:** select this checkbox to to exclude from ingestion any domains whose reputation/risk score value is lower than the specified *low maliciousness threshold*.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the DomainTools Suspicious Domains enricher timeout value is set to 1 minute.

Elasticsearch sightings enricher

The Elasticsearch sightings enricher creates sightings from matching results returned from a search in an external Elasticsearch instance.

This article describes how to configure the Elasticsearch sightings enricher parameters. To configure the general options for the Elasticsearch sightings enricher, see [Configure enrichers](#).

Elasticsearch sightings	enricher
Enricher name	Elasticsearch sightings
API endpoint	<code>http://\${elasticsearch_instance_url}:9200/\${schema_resource}</code>
Input	domain, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
Output	Creates sightings from matching results returned from a search in an external Elasticsearch instance.
Description	Searches an external Elasticsearch instance. Any hits matching the search criteria are processed to automatically generate corresponding sightings.

Configure the Elasticsearch sightings enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Elasticsearch sightings enricher.

Supported observable types:

- *domain*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *hash-sha512*
- *host*
- *ipv4*
- *ipv6*
- *uri*

Under **Parameters**, define the specific configuration options for the Elasticsearch sightings enricher:

- **ElasticSearch URL:** enter the URL pointing to the external Elasticsearch instance you want to use as a source for the enricher, including the sub-resource pointing to the **data mapping schema** (<https://www.elastic.co/guide/en/elasticsearch/guide/current/mapping-intro.html>).
Example: *http://localhost:9200/default*

In a usage scenario, you may want to obtain data from an external Elasticsearch instance that acts as a centralized log aggregator to check for correlations with the platform observables, indicators, and other entities. If it is possible to establish a relationship between Elasticsearch data and a platform entity, a sighting is automatically created.

- **Username:** enter valid user name credentials to authenticate and to receive authorization to access the resource(s).
Example: *nigeltufnel*.
- **Password:** enter valid password credentials to authenticate and to receive authorization to access the resource(s).
Example: *thesegeto11*.
- **Observable queries:** from the drop-down menu select the observable type and the corresponding observable value the rule should look for.
 - In the first input field, from the drop-down menu select the *observable type* the rule should look for.
Supported observable types:
 - *domain*
 - *hash-md5*
 - *hash-sha1*
 - *hash-sha256*
 - *hash-sha512*
 - *host*
 - *ipv4*
 - *ipv6*
 - *uri*
 - In the second input field, specify the *observable value* associated with the observable type that the rule should look for.

You can use free text, wildcards, **Elasticsearch query syntax**

(<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html>), as well as the *{kind}* and *{value}* placeholders to reference an observable type and value, respectively.

When the query executes, the placeholders take the values from the input observable key (*{kind}*) and value (*{value}*) pairs, respectively.

Example:

The **@{value}* query searches for observable values matching the input observable values it is fed at runtime.

- Click **+ Add** or **+ More** to add a new filtering option. For example, to include in the search additional key/value pairs like IP addresses, hashes, or domains.
- **Search results limit:** if you want to limit the returned search results, so that the search result entries do not exceed a predefined amount, you can set a cap here.
For example: *10*.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the Elasticsearch sightings enricher timeout value is set to *2 minutes*.

Farsight DNSDB enricher

The Farsight DNSDB enricher provides historical passive DNS information to relate domain names to the IP addresses they point to, or IPs pointing to different domains over time.

This article describes how to configure the Farsight DNSDB enricher parameters.

To configure the general options for the Farsight DNSDB enricher, see [Configure enrichers](#).

Farsight DNSDB	enricher
Enricher name	Farsight DNSDB
API endpoint	<code>https://api.dnsdb.info/{}</code>
Input	domain, host, ipv4, ipv6
Output	Enriches supported observable types with passive DNS lookup information such as the name of the domain or host name owner, or the IP address a domain or host name points to.
Description	Historical passive DNS lookup enricher. It can retrieve previous domains pointing to a specified IP address in the past, domain names hosted by a nameserver, domain names pointing to an IP network, and subdomains existing below a parent domain name.

Configure the Farsight DNSDB enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Farsight DNSDB enricher.

Supported observable types:

- *domain*
- *host*
- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the Farsight DNSDB enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
The API URL to reach the Farsight DNSDB service is `https://api.dnsdb.info/{}.`
- **API key:** contact Farsight to receive an API key for the DNSDB service, and then enter it in the corresponding input field.
- **Search results limit:** enter an integer to limit the maximum amount of returned results.
Default value: each time the enricher runs, it can return max. *1000* matches.

- **Time last seen:** enter an integer to set a starting point in the past to retrieve matches from. The number indicates the number of days in the past from the current time.
Default value: 365 (each time the enricher runs, it looks for matches up to one year old)
- Click **Save** to store your changes, or **Cancel** to discard them.

FireEye iSIGHT enricher

The FireEye iSIGHT enricher retrieves threat intelligence reports about threats and malware related to areas such as critical infrastructure, cyber crime and espionage, hacktivism, frauds, and vulnerability and exploitation.

This article describes how to configure the FireEye iSIGHT enricher parameters.

To configure the general options for the FireEye iSIGHT enricher, see [Configure enrichers](#).

FireEye iSIGHT	enricher
Enricher name	FireEye iSIGHT
API endpoint	<code>https://api.isightpartners.com/search/{}</code>
Input	asn, domain, email, email-subject, file, hash-md5, hash-sha1, hash-sha256, host, ipv4, ipv6, uri
Output	Enriches supported observable types related to the matching input observable types.
Description	Enriches platform observables with data about threat www.threatcrowd.org/s and malware related to areas such as critical infrastructure, cyber crime and espionage, hacktivism, frauds, and vulnerability and exploitation.

Configure the FireEye iSIGHT enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the FireEye iSIGHT enricher.

Supported observable types:

- *asn*
- *domain*
- *email*
- *email-subject*
- *file*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *host*
- *ipv4*
- *ipv6*
- *uri*

Under **Parameters**, define the specific configuration options for the FireEye iSIGHT enricher:

- **FireEye API URL**: the URL pointing to the API endpoint exposing the service that grants access to the incoming feed data. Contact the intel provider to subscribe to the service and to obtain this information.
The API URL to reach the FireEye iSIGHT service is [https://api.isightpartners.com/search/{}.](https://api.isightpartners.com/search/{})
- **API public key**: enter the API public key to authorize your account. To access the FireEye iSIGHT service you need to authenticate using a public and a private key.
Make sure both keys are securely stored.
- **API private key**: enter the API private key to authorize your account. To access the FireEye iSIGHT service you need to authenticate using a public and a private key.
Make sure both keys are securely stored.
- **Search results limit**: enter an integer to limit the maximum amount of returned results.
Default value: each time the enricher runs, it can return max. 1000 matches.
- **Day offset**: enter an integer to set a starting point in the past to retrieve matches from. The number indicates the number of days in the past from the current time.
Default value: 365 (each time the enricher runs, it looks for matches up to one year old)
- **ThreatScape products**: from the drop-down menu select which ThreatScape products you want to associate with the enricher, so that it can use them as data sources:
 - *Hackivism*: data source focusing on hacktivists, black-hat hackers, and terrorist groups.
 - *Enterprise*: data source focusing on corporate threats.
 - *Cyber Espionage*: data source focusing on cyber and industrial espionage.
 - *Critical Infrastructure*: data source focusing on threats to **critical infrastructure** (https://en.wikipedia.org/wiki/critical_infrastructure) such as electricity generators, water suppliers, telecom, and so on.
 - *Cyber Crime*: data source focusing on criminal activities carried out through computers and Internet.
 - *High Value Auction Fraud*: data source focusing on illicit activities targeting auctions.
 - *Vulnerability and Exploitation*: data source focusing on vulnerabilities and exploits.
- **Intelligence types**: from the drop-down menu select the intelligence report types you want to associate with the enricher, so that it can use them as data sources:
 - *Malware*: the enricher searches intelligence reports about malware.
 - *Threat*: the enricher searches intelligence reports about threats such as threat actors, strategies, tactics, techniques, campaigns, and so on.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the FireEye iSIGHT enricher timeout value is set to 1 minute.

Flashpoint AggregINT enricher

The Flashpoint AggregINT enricher provides information on a range of cyber threat data like actor information, email and IP addresses, domains, and hash files. It focuses on threat actors and criminal groups.

This article describes how to configure the Flashpoint AggregINT enricher parameters.

To configure the general options for the Flashpoint AggregINT enricher, see [Configure enrichers](#).

Flashpoint AggregINT	enricher
Enricher name	Flashpoint AggregINT
API endpoint	<code>https://endlesstunnel.info/v3</code>
Input	actor-id, domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
Output	Enriches supported observable types with information such as IP addresses, domains, host names, and hash files.
Description	Polls data from the Flashpoint API. It provides information on malware, hosts, domains, IP addresses, and hashed files. The enricher can search thematic datasets focusing on hackers, terrorist and white supremacist groups, communities in conflict, state actors involved in cyberwarfare, and CBRN (https://en.wikipedia.org/wiki/cbrn_defense) threats. It produces enrichment observables like forum name, forum room name, user name of the author of a post (as actor-id), post content, thread title, UTC date and time of a post in ISO 8601 (https://en.wikipedia.org/wiki/iso_8601) (RFC 3339) (https://tools.ietf.org/html/rfc3339) format.

Configure the Flashpoint AggregINT enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Flashpoint AggregINT enricher.
Supported observable types:
 - *actor-id*
 - *domain*
 - *email*
 - *hash-md5*
 - *hash-sha1*
 - *hash-sha256*
 - *hash-sha512*
 - *host*
 - *ipv4*
 - *ipv6*
 - *uri*

Under **Parameters**, define the specific configuration options for the Flashpoint AggregINT enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
- **Username:** enter the user name associated with the Flashpoint AggregINT account to access and consume the Flashpoint AggregINT service.
- **Password:** enter the password associated with the Flashpoint AggregINT account to access and consume the Flashpoint AggregINT service.
- **Hacker dataset:** select this checkbox to search data on hacker groups and activities.
- **Terrorist dataset:** select this checkbox to search data on terrorist groups and activities.
- **White supremacist dataset:** select this checkbox to search data on white supremacist groups and activities.
- **CBRN dataset:** select this checkbox to search data on **CBRN** (https://en.wikipedia.org/wiki/cbrn_defense) threats.
- **State actor dataset:** select this checkbox to search data on state actors, that is, individuals who act on behalf of a governmental body, and their activities.
- **Communities in conflict dataset:** select this checkbox to search data on groups and communities currently in conflict with each other.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the Flashpoint AggregINT enricher timeout value is set to *2 minutes*.

Flashpoint Blueprint enricher

The Flashpoint Blueprint enricher provides information on a range of cyber threat data like actor information, email and IP addresses, domains, and hash files. It focuses on threat actors and criminal groups.

This article describes how to configure the Flashpoint Blueprint enricher parameters.

To configure the general options for the Flashpoint Blueprint enricher, see [Configure enrichers](#).

Flashpoint Blueprint	enricher
Enricher name	Flashpoint Blueprint
API endpoint	<code>https://endlesstunnel.info/v3</code>
Input	actor-id, domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
Output	Enriches supported observable types with information such as IP addresses, domains, host names, and URLs.
Description	Polls data from the Flashpoint API. It provides information based on geolocation and IP ranges, as well as on country scope. The enricher can search thematic datasets focusing on hackers, terrorist and white supremacist groups, state actors involved in cyberwarfare, and CBRN (https://en.wikipedia.org/wiki/cbrn_defense) threats. It produces enrichment observables like city/country name or IP address hit, latitude/longitude or IP address hit, forum name and thread title related to a hit, user name uniquely matched to an IP address hit.

Configure the Flashpoint Blueprint enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Flashpoint Blueprint enricher.
Supported observable types:
 - *actor-id*
 - *domain*
 - *email*
 - *hash-md5*
 - *hash-sha1*
 - *hash-sha256*
 - *hash-sha512*
 - *host*
 - *ipv4*
 - *ipv6*
 - *uri*

Under **Parameters**, define the specific configuration options for the Flashpoint Blueprint enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
- **Username:** enter the user name associated with the Flashpoint Blueprint account to access and consume the Flashpoint Blueprint service.
- **Password:** enter the password associated with the Flashpoint Blueprint account to access and consume the Flashpoint Blueprint service.
- **Hacker dataset:** select this checkbox to search data on hacker groups and activities.
- **Terrorist dataset:** select this checkbox to search data on terrorist groups and activities.
- **White supremacist dataset:** select this checkbox to search data on white supremacist groups and activities.
- **CBRN dataset:** select this checkbox to search data on **CBRN** (https://en.wikipedia.org/wiki/cbrn_defense) threats.
- **State actor dataset:** select this checkbox to search data on state actors, that is, individuals who act on behalf of a governmental body, and related activities.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the Flashpoint Blueprint enricher timeout value is set to *2 minutes*.

Flashpoint Thresher enricher

The Flashpoint Thresher enricher provides information on a range of cyber threat data like actor information, email and IP addresses, domains, and hash files. It focuses on threat actors and criminal groups.

This article describes how to configure the Flashpoint Thresher enricher parameters.

To configure the general options for the Flashpoint Thresher enricher, see [Configure enrichers](#).

Flashpoint Thresher	enricher
Enricher name	Flashpoint Thresher
API endpoint	<code>https://endlesstunnel.info/v3</code>
Input	domain, file, hash-sha1, host, ipv4, uri
Output	Enriches supported observable types with information such as IP addresses, domains, URLs, hashes, and files.
Description	Polls data from the Flashpoint API. The enricher can search thematic datasets focusing on hackers, terrorist and white supremacist groups, and CBRN (https://en.wikipedia.org/wiki/cbrn_defense) threats. It produces enrichment observables with Flashpoint torrent thresher data.

Configure the Flashpoint Thresher enricher parameters



Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Flashpoint Thresher enricher.

Supported observable types:

- *domain*
- *file*
- *hash-sha1*
- *host*
- *ipv4*
- *uri*

Under **Parameters**, define the specific configuration options for the Flashpoint Thresher enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.

- **Username:** enter the user name associated with the Flashpoint Thresher account to access and consume the corresponding Flashpoint Thresher service.
- **Password:** enter the password associated with the Flashpoint Thresher account to access and consume the corresponding Flashpoint Thresher service.
- **Hacker dataset:** select this checkbox to search data on hacker groups and related activities.
- **Terrorist dataset:** select this checkbox to search data on terrorist groups and related activities.
- **White supremacist dataset:** select this checkbox to search data on white supremacist groups and related activities.
- **CBRN dataset:** select this checkbox to search data on **CBRN** (https://en.wikipedia.org/wiki/cbrn_defense) threats.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the Flashpoint Thresher enricher timeout value is set to *2 minutes*.

Fox-IT InTELL Portal enricher

The Fox-IT InTELL Portal enricher provides information from a range of sources, such as forums and sites that have registered potentially suspicious activity.

This article describes how to configure the Fox-IT InTELL Portal enricher parameters.

To configure the general options for the Fox-IT InTELL Portal enricher, see [Configure enrichers](#).

Fox-IT InTELL Portal	enricher
Enricher name	Fox-IT InTELL Portal
API endpoint	https://cybercrime-portal.fox-it.com/
Input	domain, hash-md5, hash-sha1, hash-sha256, host, ipv4, ipv6, uri
Output	Enriches supported observable types with relevant contextual information from forums, chats, and IRC channels.
Description	Based on Fox-IT InTELL, the portal gathers cyber threat intelligence from a range of sources, such as forums and sites that have registered potentially suspicious activity.

Configure the Fox-IT InTELL Portal enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Fox-IT InTELL Portal enricher.

Supported observable types:

- *domain*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *host*
- *ipv4*
- *ipv6*
- *uri*

Under **Parameters**, define the specific configuration options for the Fox-IT InTELL Portal enricher:

- **Fox-IT InTELL portal URL** : the URL pointing to the API endpoint exposing the service that grants access to the enricher data. Contact the intel provider to subscribe to the service and to obtain this information.

- **SSL certificate file path:** enter the path to the locally stored *.pem* or *.crt* SSL certificate you obtain from Fox-IT after subscribing to InTELL.
- **SSL key file path:** enter the path to the locally stored *.pem* or *.key* SSL private key related to the SSL certificate.
- **Username:** enter the user name associated with the Fox-IT InTELL Portal account to access and consume the InTELL service.
- **Password:** enter the password associated with the Fox-IT InTELL Portal account to access and consume the InTELL service.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the Fox-IT InTELL Portal enricher timeout value is set to *2 minutes*.

Intel 471 enricher

The Intel 471 enricher provides information on compromised IP addresses, domains, URLs, and emails, as well as first-hand information about threat actors and groups.

This article describes how to configure the Intel 471 enricher parameters.
To configure the general options for the Intel 471 enricher, see [Configure enrichers](#).

Intel 471	enricher
Enricher name	Intel 471
API endpoint	https://api.intel471.com/v1/
Input	actor-id, domain, email, hash-md5, hash-sha256, host, ipv4, ipv6, uri
Output	Enriches supported observable types with data focusing on threat actor information.
Description	Besides data on compromised IP addresses, domains, URLs, and emails, Intel 471 focuses on providing first-hand information related to threat actors and groups.

Configure the Intel 471 enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Intel 471 enricher.
Supported observable types:
 - *actor-id*
 - *domain*
 - *email*
 - *hash-md5*
 - *hash-sha256*
 - *host*
 - *ipv4*
 - *ipv6*
 - *uri*

Under **Parameters**, define the specific configuration options for the Intel 471 enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.

The current API endpoint exposing the Intel 471 service is <https://api.intel471.com/v1/>

- **API key**: contact Intel 471 to receive an API key, and then enter it in the corresponding input field.
- **Email**: enter the email address associated with the Intel 471 account to access and consume the Intel 471 API service.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the Intel 471 enricher timeout value is set to *2 minutes*.

MISP API enricher

The MISP API enricher retrieves data from a source MISP instance. It returns enrichment information on indicators of compromise. Based on the enricher criteria and the retrieved content, ingested data is stored to the platform as enrichment observables, entities, and relationships.

This article describes how to configure the MISP API enricher parameters.

To configure the general options for the MISP API enricher, see [Configure enrichers](#).

MISP API	enricher
Enricher name	MISP API
API endpoint	<code>http://\${misp_instance_url}/</code>
Input	actor-id, bank-account, card, country, domain, email, email-subject, file, hash-md5, hash-sha1, hash-sha256, hash-sha512, ipv4, ipv6, malware, name, netname, organization, person, port, registrar, telephone, uri, yara
Output	Enriches supported observables and entities with information on indicators of compromise. Searches a source MISP instance for MISP events containing the values of the selected observable types. Generates enrichment observables, entities, and relationships based on the retrieved data.
Description	Enriches supported observables and entities with matching MISP event information.

Limitations

The native ingestion of MISP events is a significant improvement with respect to STIX-based MISP object ingestion. The current approach provides more context, and it produces a proper data hierarchy in the platform.

Keep in mind that there is a likely risk of not receiving all information, or of missing updates, due to limitations in this integration. Future platform releases will address these constraints. It is very likely that upon installing an upgraded release of the MISP integration, it will be necessary to recreate the data feed.

The following list includes all known limitations related to the current implementation.

- **Not all MISP properties** (<http://www.misp-project.org/datamodels/#types>) **and objects** (<http://www.misp-project.org/objects.html>) have been mapped to corresponding entity and observable data types in the platform.

At the moment, and after consulting with MISP Project team members, we implemented the most commonly used constructs.

- It is not yet possible to share MISP data to other MISP instances.

MISP allows data sharing across MISP instances, where events can evolve independently of the original source. When evolved events with same parent events are ingested from more than one source, the platform deduplicates them, and it ingests only the latest version, without performing any data merging.

- It is not yet possible to configure multiple MISP data sources.
- No special handling in place for proposals.

Configure the MISP API enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the MISP API enricher.

Supported observable types:

- *actor-id*
- *bank-account*
- *card*
- *country*
- *domain*
- *email*
- *email-subject*
- *file*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *hash-sha512*
- *ipv4*
- *ipv6*
- *malware*
- *name*
- *netname*
- *organization*
- *person*
- *port*
- *registrar*
- *telephone*
- *uri*
- *yara*

Under **Parameters**, define the specific configuration options for the MISP API enricher:

- **MISP API URL:** the URL pointing to the API endpoint exposing the service that grants access to the source MISP instance.
- **MISP API key:** contact the administrator of the source MISP instance to request an authentication MISP API key, and then enter it in the corresponding input field.

OpenDNS OpenResolve enricher

The Cisco OpenDNS OpenResolve enricher provides reverse-DNS lookup information.

This article describes how to configure the OpenDNS OpenResolve enricher parameters.

To configure the general options for the OpenDNS OpenResolve enricher, see [Configure enrichers](#).

Cisco OpenDNS OpenResolve	enricher
Enricher name	OpenDNS OpenResolve
API endpoint	<code>http://api.openresolve.com/{}/{} </code>
Input	domain, host, ipv4, ipv6
Output	Enriches supported observable types with reverse-DNS lookup information.
Description	OpenResolve by OpenDNS offers a REST API to use DNS resolvers and to retrieve reverse-DNS lookup information.

Configure the OpenDNS OpenResolve enricher parameters



Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the OpenDNS OpenResolve enricher.
Supported observable types:

- *domain*
- *host*
- *ipv4*
- *ipv6*

The OpenDNS OpenResolve enricher has no specific parameters to configure.

- To modify the general options for the enricher, click **Edit**.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the OpenDNS OpenResolve enricher timeout value is set to *2 minutes*.

PassiveTotal IP/Domain enricher

The PassiveTotal IP/Domain enricher provides additional information for the queried IP address or domain name.

This article describes how to configure the PassiveTotal IP/Domain enricher parameters.

To configure the general options for the PassiveTotal IP/Domain enricher, see [Configure enrichers](#).

PassiveTotal IP/Domain	enricher
Enricher name	PassiveTotal IP/Domain
API endpoint	https://api.passivetotal.org/v2
Input	domain, host, ipv4, ipv6
Output	Enriches supported observable types with enrichment (https://passivetotal.readthedocs.io/en/latest/api.html#enrichment-request) information.
Description	Polls data from the PassiveTotal API (https://api.passivetotal.org/api/docs/#api-enrichment-getv2enrichmentquery). It provides additional context related to the queried IP address or domain name. For example, it returns domain name, any sub-domains, inet details, autonomous system number (ASN) (https://en.wikipedia.org/wiki/autonomous_system_(internet)), as well as geolocation information. Analysts can query the returned data to look for further connections that may be relevant during an investigation.

Configure the PassiveTotal IP/Domain enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the PassiveTotal IP/Domain enricher.
Supported observable types:

- *domain*
- *host*
- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the PassiveTotal IP/Domain enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.

- **API key**: contact PassiveTotal to receive an API key, and then enter it in the corresponding input field.
- **Email**: enter the email address associated with the PassiveTotal IP/Domain account to access and consume the PassiveTotal IP/Domain API service.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the PassiveTotal IP/Domain enricher timeout value is set to *2 minutes*.

PassiveTotal Malware enricher

The PassiveTotal Malware enricher provides malware information related to the queried host or domain, such as malware hash and collection date.

This article describes how to configure the PassiveTotal Malware enricher parameters.

To configure the general options for the PassiveTotal Malware enricher, see [Configure enrichers](#).

PassiveTotal Malware	enricher
Enricher name	PassiveTotal Malware
API endpoint	https://api.passivetotal.org/v2
Input	domain, host
Output	Enriches supported observable types with malware enrichment (https://passivetotal.readthedocs.io/en/latest/api.html?highlight=malware#enrichment-request) information .
Description	Polls data from the PassiveTotal API (https://api.passivetotal.org/api/docs/#api-enrichment-getv2enrichmentmalwarequery). It returns malware information related to the queried host or domain, such as malware hash — hash-md5, hash-sha1, hash-sha256, hash-sha512 — and collection date. The returned malware entries are also tagged with the following metadata: <code>enrichment_extracts.meta.classification: bad</code> — it corresponds to the value you set under Data configuration > Rules > Observable > + > Action > Mark as malicious ; <code>enrichment_extracts.meta.confidence: low</code> — it corresponds to the value you set under Data configuration > Rules > Observable > + > Confidence > Malicious - Low confidence .

Configure the PassiveTotal Malware enricher parameters



Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the PassiveTotal Malware enricher.

Supported observable types:

- *domain*
- *host*

Under **Parameters**, define the specific configuration options for the PassiveTotal Malware enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
- **API key:** contact PassiveTotal to receive an API key, and then enter it in the corresponding input field.

- **Email:** enter the email address associated with the PassiveTotal Malware account to access and consume the PassiveTotal Malware API service.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the PassiveTotal Malware enricher timeout value is set to *2 minutes*.

PassiveTotal Passive DNS enricher

The PassiveTotal Passive DNS enricher provides historical information to enable cross-referencing IP addresses to the corresponding DNS domain names over time.

This article describes how to configure the PassiveTotal Passive DNS enricher parameters.

To configure the general options for the PassiveTotal Passive DNS enricher, see [Configure enrichers](#).

PassiveTotal Passive DNS	enricher
Enricher name	PassiveTotal Passive DNS
API endpoint	https://api.passivetotal.org/v2
Input	domain, host, ipv4, ipv6
Output	Enriches supported observable types with passive DNS (https://www.riskiq.com/products/learn-threat-research-and-analysis/) information.
Description	Polls data from the PassiveTotal API (https://api.passivetotal.org/api/docs/#api-dns-getv2dnspassivequery). It provides historical information that allows cross-referencing IP addresses to the corresponding DNS domain names over time. Analysts can examine how a domain name resolves to different IP addresses over time. They can then query the IP addresses to retrieve more domain names that may be relevant in an investigation.

Configure the PassiveTotal Passive DNS enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the PassiveTotal Passive DNS enricher.
Supported observable types:

- *domain*
- *host*
- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the PassiveTotal Passive DNS enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
- **API key:** contact PassiveTotal to receive an API key, and then enter it in the corresponding input field.

- **Email:** enter the email address associated with the PassiveTotal Passive DNS account to access and consume the PassiveTotal Passive DNS API service.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the PassiveTotal Passive DNS enricher timeout value is set to *2 minutes*.

PassiveTotal Whois enricher

The PassiveTotal Whois enricher provides information about individuals or entities associated with an IP address or a domain name, as well as geolocation details.

This article describes how to configure the PassiveTotal Whois enricher parameters.

To configure the general options for the PassiveTotal Whois enricher, see [Configure enrichers](#).

PassiveTotal Whois	enricher
Enricher name	PassiveTotal Whois
API endpoint	https://api.passivetotal.org/v2
Input	domain, host, ipv4, ipv6
Output	Enriches supported observable types with whois (https://www.riskiq.com/products/learn-threat-research-and-analysis/) information.
Description	Polls data from the PassiveTotal API (https://api.passivetotal.org/api/docs/#api-whois-getv2whoisquery). It provides information about individuals or entities associated with an IP address or a domain name, as well as geolocation details. Analysts can retrieve registrar, organization, country, city, street, telephone, and email details. They can then use these details to run further queries to obtain, for example, more domain names associated with the same individual or the same company.

Configure the PassiveTotal Whois enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the PassiveTotal Whois enricher.
Supported observable types:

- *domain*
- *host*
- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the PassiveTotal Whois enricher:

- **API URL:** the URL pointing to the API endpoint exposing the service that grants access to the enricher data source. Contact the intel provider to subscribe to the service and to obtain this information, as well as any required authentication and authorization credentials.
- **API key:** contact PassiveTotal to receive an API key, and then enter it in the corresponding input field.

- **Email:** enter the email address associated with the PassiveTotal Whois account to access and consume the PassiveTotal Whois API service.
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the PassiveTotal Whois enricher timeout value is set to *2 minutes*.

Proofpoint Email Threat enricher

The Proofpoint Email Threat enricher uses input data such as email subjects, domain names, hashes, and IP addresses to return information on email threats such as phishing, spoofing, and email malware. The ingested Proofpoint data is stored to the platform as email threat indicators.

This article describes how to configure the Proofpoint Email Threat enricher parameters.

To configure the general options for the Proofpoint Email Threat enricher, see [Configure enrichers](#).

Proofpoint Email Threat	enricher
Enricher name	Proofpoint Email Threat
API endpoint	<code>https://api.emaildefense.proofpoint.com/v1</code>
Input	domain, email-subject, hash-md5, ipv4, uri
Output	Enriches supported observables and entities with information on email threats. Generates email threat indicators containing observables that represent the ingested data, as well as relationships between the enriched entities and the enrichment data. The output includes, when available, the brand a spoofing message uses to trick users, the message subject line, source IP address, source host name, and source email address, as well as any MD5 file hashes, if the malicious email contains attachments.
Description	Enriches supported observables and entities with information on email threats such as phishing, spoofing, email malware, and impostor email/fraudulent messages.

Configure the Proofpoint Email Threat enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Proofpoint Email Threat enricher.

Supported observable types:

- *domain*
- *email-subject*
- *hash-md5*
- *ipv4*
- *uri*

Under **Parameters**, define the specific configuration options for the Proofpoint Email Threat enricher:

- **API key:** contact Proofpoint to receive an API key, and then enter it in the corresponding input field.

- **API URL**: the URL pointing to the API endpoint exposing the service that grants access to the enricher data. Contact the intel provider to subscribe to the service and to obtain this information.

The current API endpoint exposing the Proofpoint Email Threat service is *<https://api.emaildefense.proofpoint.com/v1>*.

- **Likely impact lower threshold**: enter an integer value to assess the maliciousness confidence level of detected email threats.

This value sets the minimum maliciousness confidence value to flag a potential email threat as *somewhat likely* to be malicious.

The lower threshold value needs to be smaller than the highest threshold value.

- Enter a value between *0* and *100*.
- Default value: *60*.

- **Likely impact higher threshold**: enter an integer value to assess the maliciousness confidence level of detected email threats.

This value sets the minimum maliciousness confidence value to flag a potential email threat as *very likely* to be malicious.

The higher threshold value needs to be bigger than the lower threshold value.

- Enter a value between *0* and *100*.
- Default value: *90*.

About impact thresholds

- Incoming enrichment observables with a *lower* value than the one you set in **Likely impact lower threshold** are automatically flagged as **Malicious - Low confidence**.
- Incoming enrichment observables whose likely impact value is *bigger* than the *lower threshold* and *smaller* than the *higher threshold* values are automatically flagged as **Malicious - Medium confidence**.
- Incoming enrichment observables with a *higher* value than the one you set in **Likely impact higher threshold** are automatically flagged as **Malicious - High confidence**.

You can use impact threshold values to automatically filter enricher output data, and to instrument external systems and devices, such as a SIEM, to execute follow-up actions. For example, add all high confidence maliciousness email addresses to a blocklist.

- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the Proofpoint Email Threat enricher timeout value is set to *2 minutes*.

PyDat enricher

The PyDat enricher provides whois, including historical whois, and passive DNS lookup information.

This article describes how to configure the PyDat enricher parameters.

To configure the general options for the PyDat enricher, see [Configure enrichers](#).

PyDat	enricher
Enricher name	PyDat
API endpoint	<code>http://\${pydat_instance_url}:8000/</code> (example)
Input	domain, ipv4, ipv6
Output	Enriches supported observable types with whois data, current IP resolution and passive DNS information.
Description	PyDat (https://github.com/mitrecnd/whodat#pydat) is installed locally, and it can work together with an Elasticsearch instance (https://github.com/mitrecnd/whodat/tree/master/pydat#pydat-with-elasticsearch) to provide whois, including historical whois, and passive DNS lookup information. Analysts can retrieve name, organization, country, city, street, ZIP code, telephone, and email details.

Configure the PyDat enricher parameters

- ✓ Input fields marked with an asterisk are required.

You need to install and set up PyDat locally. The product does not work outside a local network. You need to configure the host before you can access PyDat features through the API endpoint. See also:

- **Mitre blog on PyDat**
(<http://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/using-whois-and-passive-dns-for-intelligence>)
- **PyDat GitHub repo** (<https://github.com/mitrecnd/whodat>)

Under **Parameters**, define the specific configuration options for the PyDat enricher:

- **API URL**: the URL allowing access to the local **PyDat** (<https://github.com/mitrecnd/whodat#pydat-api>) instance.
Example: `http://${pydat_instance_url}:8000/`
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the PyDat enricher timeout value is set to *2 minutes*.

Recorded Future enricher

The Recorded Future enricher enables you to tap into the data stream generated by the Recorded Future Temporal Analytics Engine to retrieve search results potentially malicious IPs, domains, email addresses, and hashes related to the input observable types, along with their risk scores to automatically flag domains with an appropriate maliciousness confidence level.

This article describes how to configure the Recorded Future enricher parameters.

To configure the general options for the Recorded Future enricher, see [Configure enrichers](#).

Recorded Future	enricher
Enricher name	Recorded Future
API endpoint	<code>https://api.recordedfuture.com/live/sc/entity/{}</code>
Input	domain, hash-md5, hash-sha1, hash-sha256, hash-sha256, ipv4, ipv6
Output	Enriches supported observable types with pattern matching search results produced by the Recorded Future Temporal Analytics Engine.
Description	The enricher returns additional data such as IPs, domains, email addresses, and hashes related to the submitted observables in the specified types, as well as maliciousness confidence levels based on the retrieved risk scores.

Configure the Recorded Future enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Recorded Future enricher.

Supported observable types:

- *domain*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *hash-sha256*
- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the Recorded Future enricher:

- **API key:** contact Recorded Future to receive an API key, and then enter it in the corresponding input field.

Maliciousness confidence rating is based on the Recorded Future risk scoring, where *0* means *no current evidence of risk*, whereas *99* means *very malicious*:

- **Low maliciousness threshold:** analyzed supported observables with a higher Recorded Future risk score than the value defined here are flagged as **Malicious - Low confidence**.
After completing the analysis, enriched observables with a *higher* risk score than the *low maliciousness threshold* and lower than the medium and high maliciousness thresholds are flagged as **Malicious - Low confidence**.
 - Enter a value between *0* and *99*.
 - Default value: *5*.
- **Medium maliciousness threshold:** analyzed supported observables with a higher Recorded Future risk score than the value defined here are flagged as **Malicious - Medium confidence**.
After completing the analysis, enriched observables with a *higher* risk score than the *medium maliciousness threshold* and lower than the high maliciousness threshold are flagged as **Malicious - Medium confidence**.
 - Enter a value between *0* and *99*.
 - Default value: *24*.
- **High maliciousness threshold:** analyzed supported observables with a higher Recorded Future risk score than the value defined here are flagged as **Malicious - High confidence**.
After completing the analysis, enriched observables with a *higher* risk score than the *high maliciousness threshold* are flagged as **Malicious - High confidence**.
 - Enter a value between *0* and *99*.
 - Default value: *65*.
- Click **Save** to store your changes, or **Cancel** to discard them.

RIPEstat GeoIP enricher

The RIPEstat GeoIP enricher provides geolocation information related to the queried IP addresses.

This article describes how to configure the RIPEstat GeoIP enricher parameters.

To configure the general options for the RIPEstat GeoIP enricher, see [Configure enrichers](#).

RIPEstat GeoIP	enricher
Enricher name	RIPEstat GeoIP
API endpoint	https://stat.ripe.net/data/geoloc/{} (Geoloc (https://stat.ripe.net/docs/data_api#geoloc)
Input	ipv4, ipv6
Output	Enriches supported observable types with geolocation information related to IP addresses: coordinates, country, and city.
Description	Geolocation IP information from the RIPEstat web-based interface (Data API (https://stat.ripe.net/docs/data_api)), including latitude, longitude, country, and city.

Configure the RIPEstat GeoIP enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the RIPEstat GeoIP enricher.

Supported observable types:

- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the RIPEstat GeoIP enricher:

- **API URL:** the basic URL allowing access to the **RIPEstat Data API** (https://stat.ripe.net/docs/data_api). The value is: *https://stat.ripe.net/data*.
- Click **Save** to store your changes, or **Cancel** to discard them.

i By default, the RIPEstat GeoIP enricher timeout value is set to *2 minutes*.

RIPEstat Whois enricher

The RIPEstat Whois enricher provides whois information related to the queried IP addresses.

This article describes how to configure the RIPEstat Whois enricher parameters.

To configure the general options for the RIPEstat Whois enricher, see [Configure enrichers](#).

RIPEstat Whois	enricher
Enricher name	RIPEstat Whois
API endpoint	https://stat.ripe.net/data/whois/{} (Whois (https://stat.ripe.net/docs/data_api#whois)
Input	ipv4, ipv6
Output	Enriches supported observable types with whois information related to IP addresses.
Description	Whois information from the RIPEstat web-based interface (Whois REST API (https://github.com/ripe-ncc/whois/wiki/whois-rest-api)), including inet number, name, organization, country, city, street, and telephone.

Configure the RIPEstat Whois enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the RIPEstat Whois enricher.

Supported observable types:

- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the RIPEstat Whois enricher:

- **API URL:** the basic URL allowing access to the **RIPEstat Data API** (https://stat.ripe.net/docs/data_api). The value is: *https://stat.ripe.net/data*.
- Click **Save** to store your changes, or **Cancel** to discard them.

i By default, the RIPEstat Whois enricher timeout value is set to *2 minutes*.

Shodan enricher

The Shodan enricher uses input data such as country and city names, organization and personal names, ZIP codes, email addresses, and so on to return a list of matching IP addresses corresponding to your Internet-connected devices, along with location and user details.

This article describes how to configure the Shodan enricher parameters.
To configure the general options for the Shodan enricher, see [Configure enrichers](#).

Shodan	enricher
Enricher name	Shodan
API endpoint	https://api.shodan.io/shodan/
Input	asn, city, company, country, country-code, domain, email, geo-lat, geo-long, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, organization, person, port, postcode, uri
Output	Enriches supported observable types with the following information, when available: country name, city name, ZIP code, longitude, latitude, organization name, host name, IP address, open ports and services related to input IP addresses.
Description	Shodan matches input data such as email addresses, physical locations, company or personal names, to related IP addresses. It enables discovering which devices in your system or environment are connected to the Internet, where they are located, and who uses them.

Configure the Shodan enricher parameters



Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Shodan enricher.

Supported observable types:

- *asn*
- *city*
- *company*
- *country*
- *country-code*
- *domain*
- *email*
- *geo-lat*
- *geo-long*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *hash-sha512*
- *host*
- *ipv4*
- *ipv6*
- *organization*
- *person*
- *port*
- *postcode*
- *uri*

Under **Parameters**, define the specific configuration options for the Shodan enricher:

- **API user name:** sign up and subscribe to the service to obtain the required API user name and API key credentials to access the API endpoint exposing the service.
- **API key:** contact Shodan and create a user account to receive an API key; then enter it in the corresponding input field.
- Click **Save** to store your changes, or **Cancel** to discard them.



Although the Shodan enricher supports the *ipv6* observable type as a valid input, at the moment the Shodan API does not return any results for *ipv6* searches.

Splunk sightings enricher

The Splunk sightings enricher searches the indexes in the specified Splunk instance. Matching data is extracted and saved to the platform as sightings.

This article describes how to configure the Splunk sightings enricher parameters.

To configure the general options for the Splunk sightings enricher, see [Configure enrichers](#).

Splunk sightings	enricher
Enricher name	Splunk sightings
API endpoint	<code>http://\${splunk_instance_url}:8089/</code> (example)
Input	domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
Output	Creates sightings for matching input observables, based on the search result items retrieved in the specified Splunk instance.
Description	Based on the search queries defined in the enricher, the enricher looks for matching data in the specified Splunk instance. Matching data is extracted and saved to the platform as sightings.

Configure the Splunk sightings enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the Splunk sightings enricher.

Supported observable types:

- *domain*
- *email*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *hash-sha512*
- *host*
- *ipv4*
- *ipv6*
- *uri*

Under **Parameters**, define the specific configuration options for the Splunk sightings enricher:

- **Splunk URL:** enter the URL pointing to the Splunk instance you want the enricher to search for matching data. The default management port listening to the *splunkd* daemon is port *8089*: make sure this **port is open** (<https://answers.splunk.com/answers/58888/what-are-the-ports-that-i-need-to-open.html>). Example: *https://splunk.instance.com:8089*
- **Username:** enter the user name associated with a valid Splunk account to access the Splunk instance you want to use as a data source for the enricher.
- **Password:** enter the password associated with a valid Splunk account to access the Splunk instance you want to use as a data source for the enricher.
- **Observable queries:** you define observable queries as key/value pairs. Each query targets a specific observable data type. You can add as many queries as you need.
 - The first field holds the query key: from the drop-down menu select the observable data type you want to look for in the specified Splunk instance.
Supported observable types:
 - *domain*
 - *email*
 - *hash-md5*
 - *hash-sha1*
 - *hash-sha256*
 - *hash-sha512*
 - *host*
 - *ipv4*
 - *ipv6*
 - *uri*
 - The second field holds the query value: define the value of the observable type you want the enricher to look for in the specified Splunk instance.

Input values need to be in a valid format for the selected observable type.

Allowed formats:

- Literal values; for example a domain name, an IP address, or an email address;
- **Wildcards** (<https://docs.splunk.com/documentation/splunk/latest/search/wildcards>);
- **Boolean expressions**
(<https://docs.splunk.com/documentation/splunk/latest/search/booleanexpressions>);
- **Field expressions**
(<https://docs.splunk.com/documentation/splunk/latest/search/fieldexpressions>);
- **SPL** (<https://docs.splunk.com/splexicon:spl>) and **regex**
(<https://docs.splunk.com/documentation/splunk/latest/search/splandregularexpressions>).
- *{kind}* and *{value}* placeholders to reference an observable type and value, respectively.

When the query executes, the placeholders take the values from the input observable key (*{kind}*) and value (*{value}*) pairs, respectively.

Example:

The **@{value}* query searches for observable values matching the input observable values it is fed at runtime.

- Click **+ Add** or **+ More** to add new rows as needed, where you can enter additional queries targeting other observable data types.

- **Search results limit:** enter an integer if you want to limit the number of results the search returns for each query.
- Click **Save** to store your changes, or **Cancel** to discard them.

Search result matches generate sightings that are saved to the platform.
Each sighting includes the following information:

- A unique ID;
- A URL pointing to the Splunk instance data source;
- A URL with the query that retrieved the data;
- Details about the sighted observable.

For example, a Splunk index reference, the source log the data was found in, a timestamp, and any raw response data, if available.



By default, the Splunk sightings enricher timeout value is set to *2 minutes*.

SpyCloud Breach Data enricher

The SpyCloud Breach Data enricher uses input data such as IP addresses, domain names, email addresses and user access credentials to return incident and security breach information that is processed as incident entities, related observables, and enrichment observables.

This article describes how to configure the SpyCloud Breach Data enricher parameters.

To configure the general options for the SpyCloud Breach Data enricher, see [Configure enrichers](#).

SpyCloud Breach Data	enricher
Enricher name	SpyCloud Breach Data
API endpoint	<code>https://api.spycloud.io/sp-v1/breach</code>
Input	domain, email, handle, ipv4, ipv6
Output	Enriches supported observable types and incident entities with account takeover (ATO) and security breach details. Generates Cybox observables, related observables, and CIQ objects. Supported output observables: <i>city</i> , <i>country-code</i> , <i>domain</i> , <i>email</i> , <i>geo-lat</i> , <i>geo-long</i> , <i>handle</i> , <i>host</i> , <i>ipv4</i> , <i>ipv6</i> , <i>organization</i> , <i>person</i> , <i>postcode</i> , <i>telephone</i> .
Description	Enriches supported observable types, as well as incident entities, with information about account takeover and security breaches, such as a description of the incident, targeted organization(s), type(s) and number of stolen records — for example, leaked or stolen assets or user access credentials — as well as when the records were made public or put for sale on the Internet. Each entry in a breach description — for example, an entry of a breached database containing login credentials (user name, password, and email address) — is a breach record.

Configure the SpyCloud Breach Data enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the SpyCloud Breach Data enricher.


Supported observable types:

- *domain*
- *email*
- *handle*
- *ipv4*
- *ipv6*

Under **Parameters**, define the specific configuration options for the SpyCloud Breach Data enricher:

- **SpyCloud API URL** : the URL pointing to the API endpoint exposing the service that grants access to the enricher data. Contact the intel provider to subscribe to the service and to obtain this information.

The current API endpoint exposing the SpyCloud Breach Data service is <https://api.spycloud.io/sp-v1/breach>.

- **SpyCloud API key** : contact SpyCloud and create a user account to receive an API key; then enter it in the corresponding input field.
- **Enrich incidents starting from** : click the  icon and select a date in the past. The enricher searches the SpyCloud breach catalog for relevant data starting from the specified date until the current time.

If no date in the past is specified, the enricher defaults to searching for relevant data up to 60 days/2 months back from the current time.

Format: *dd.MM.yyyy hh:mm:ss*

Example: *07.02.2017 23:00:00*

- Click **Save** to store your changes, or **Cancel** to discard them.

The SpyCloud Breach Data enricher augments existing incident entities; if no matching incidents exist in the platform, it creates new incidents from the retrieved enrichment data.

When it retrieves personal data related to an actor or a victim, it checks if such information already exists in the platform; otherwise, it creates **CIQ** (https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ciq) **3.0** (<http://docs.oasis-open.org/ciq/v3.0/specs/ciq-specs-v3.html>) -compliant *identity* type objects.

ThreatCrowd enricher

The ThreatCrowd enricher returns suspicious and potentially malicious domains, IP addresses, email addresses, file hashes, and antivirus detections, so that you can explore relationships between events, actors, and targets.

This article describes how to configure the ThreatCrowd enricher parameters.

To configure the general options for the ThreatCrowd enricher, see [Configure enrichers](#).

ThreatCrowd	enricher
Enricher name	ThreatCrowd
API endpoint	<code>https://www.threatcrowd.org/{}</code>
Input	domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, malware
Output	Enriches supported observable types with suspicious and potentially malicious domains, IP addresses, email addresses, file hashes, and antivirus detections.
Description	Returns suspicious and potentially malicious domains, IP addresses, email addresses, file hashes, and antivirus detections, so that you can explore relationships between events, actors, and targets.

Configure the ThreatCrowd enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the ThreatCrowd enricher.

Supported observable types:

- *domain*
- *email*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *hash-sha512*
- *host*
- *ipv4*
- *ipv6*
- *malware*

Under **Parameters**, define the specific configuration options for the ThreatCrowd enricher:

- **Time last seen:** enter an integer to set a starting point in the past to retrieve matches from. The number indicates the number of days in the past from the current time.
Default value: 365 (each time the enricher runs, it looks for matches up to one year old)
- Click **Save** to store your changes, or **Cancel** to discard them.



By default, the ThreatCrowd enricher timeout value is set to *1 minute*.

Unshorten-URL enricher

The Unshorten-URL polls the specified URL shortener services to return the resolved original URLs corresponding to the submitted shortened ones.

This article describes how to configure the Unshorten-URL enricher parameters.

To configure the general options for the Unshorten-URL enricher, see [Configure enrichers](#).

RIPEstat GeolP	enricher
Enricher name	Unshorten-URL
API endpoint	<code>https://unshorten.me/s/{}</code>
Input	uri
Output	Original URL corresponding to the submitted shortened one.
Description	It takes shortened URL as an input, and it returns the corresponding resolved original URLs, which can then be analyzed in the platform to discover relationships with other entities.

Configure the Unshorten-URL enricher parameters

✓ Input fields marked with an asterisk are required.

- **Observable types:** select the observable type representing the shortened URLs that the enricher submits to the specified services.

Supported observable types:

- *uri*

Under **Parameters**, define the specific configuration options for the Unshorten-URL enricher:

- **Providers:** enter one or more URL shortener services to use with the enricher.

Separate multiple URL shortener services with either a comma or a white space.

Example: *bit.ly,goo.gl,tinyurl.com*, or *bit.ly goo.gl tinyurl.com*

You do not need to prefix the domains with the transmission protocol. If included, *http://* or *https://* is stripped at runtime.

- Click **Save** to store your changes, or **Cancel** to discard them.

VirusTotal enricher

The VirusTotal enricher provides information on malware, domains (passive DNS) and IP addresses.

This article describes how to configure the VirusTotal enricher parameters.

To configure the general options for the VirusTotal enricher, see [Configure enrichers](#).

VirusTotal	enricher
Enricher name	VirusTotal
API endpoint	<code>https://www.virustotal.com/vtapi/v2/{}</code>
Input	domain, hash-md5, hash-sha1, hash-sha256, ipv4, ipv6, uri
Output	Enriches supported observable types with maliciousness confidence level information.
Description	Polls data from the VirusTotal API. It provides information on malware, domains (passive DNS) and IP addresses. Submitted data is checked against 60+ antimalware products, resulting in a detection ratio output and additional metadata information, when available.

Configure the VirusTotal enricher parameters



Input fields marked with an asterisk are required.

- **Observable types:** select one or more observable types you want to enrich with data retrieved through the VirusTotal enricher.
Supported observable types:

- *domain*
- *hash-md5*
- *hash-sha1*
- *hash-sha256*
- *ipv4*
- *ipv6*
- *uri*

Under **Parameters**, define the specific configuration options for the VirusTotal enricher:

- **API key: sign up** (<https://www.virustotal.com/en/documentation/public-api/#getting-started>) to the VirusTotal community to automatically be assigned a personal API key to access the VirusTotal public API, and then enter it in this field.
- **Scan URLs:** select this checkbox to to **submit URLs** (<https://www.virustotal.com/en/documentation/public-api/#scanning-urls>) to VirusTotal.

- **Scan files:** select this checkbox to to **submit files/file hashes** (<https://www.virustotal.com/en/documentation/public-api/#scanning-files>) to VirusTotal. File hashes are embedded inside entities as raw artifacts.
- **Max low confidence infection rate:** you can set an *upper threshold* to automatically flag enriched observables with a *low confidence* value.
After completing the sample analysis, enriched observables with a *lower* detection ratio than the specified value are flagged as **Malicious - Low confidence**.
 - Enter a numeric value between 0.1 and 0.9 — that is, $0 < value < 1$.
 - Default value: 0.3.
- **Min high confidence infection rate:** you can set a *bottom threshold* to automatically flag enriched observables with a *high confidence* value.
After completing the sample analysis, enriched observables with a *higher* detection ratio than the specified value are flagged as **Malicious - High confidence**.
 - Enter a numeric value between 0.1 and 0.9 — that is, $0 < value < 1$.
 - Default value: 0.6.
- Enriched observables with a detection ratio falling in the range defined by **Max low confidence infection rate** (range lower limit) and **Min high confidence infection rate** (range upper limit) are flagged as **Malicious - Medium confidence**.
- Click **Save** to store your changes, or **Cancel** to discard them.

About the confidence infection rate

$VirusTotal\ positives / VirusTotal\ engines = confidence\ infection\ rate$

To calculate the confidence infection rate value, the platform divides the number of positives — that is, infected or malicious results — the VirusTotal sample analysis returns by the total number of engines VirusTotal uses to perform the analysis.



Automatic flagging with high/low confidence maliciousness through VirusTotal supports only the following observable types:

- *hash-sha1*
- *hash-sha256*
- *hash-md5*
- *uri*

By default, the VirusTotal enricher timeout value is set to 2 minutes.

Create custom enrichers

Implement custom extensions to integrate EclecticIQ Platform with external intel providers and data sources through incoming feeds and enrichers, as well as to publish platform intel downstream in your prevention and detection toolchain.

About extensions

EclecticIQ Platform integrates with many external prevention/detection solutions and intel providers. It can exchange information through feeds, retrieve data through enrichers, and it can communicate with third-party systems through its API and ad-hoc apps that implement interoperability with specific products such as Splunk and IBM QRadar.

EclecticIQ Platform ships with out-of-the-box, ready-to-use enrichers to augment cyber threat intel with observables providing additional context. It also includes a web-based UI to create incoming and outgoing feeds, as needed.

Besides the default feeds and enrichers, you can create and implement your own. Custom feeds and enrichers implemented by third-parties other than EclecticIQ are called extensions, since they extend the platform native feature set. You can create extensions to implement additional transport types or content types for incoming or outgoing feeds, as well as new enrichers to poll data from specific intel providers.

Create enricher extensions

Before getting your hands dirty, have a look at the main steps to create an enricher extension from a boilerplate:

- Download, clone, or copy the **eclecticiq-extension-example** (<https://github.com/eclecticiq/platform-extensions/tree/master/eclecticiq-extension-example>) **extension**.
- Import dependencies.
- Create a JSON schema for the UI, if your enricher extension features a UI.
- Set a schema definition for validation.
- Define the enricher type and its instantiation through decorator functions.
- Include the necessary logic to configure the enricher behavior:
 - Define the tasks the enricher should execute;
 - Define the extract types you want the enricher to look for and retrieve.
 - Configure appropriate transport and content types to handle data formats and retrieval.
- Restart Supervisor, so that all managed processes can configure the newly added extension in **Data configuration > Enrichers**.
- Enable the extension.
- Initialize the extension by running the fixtures (applies only to enricher extensions).

Prepare the boilerplate

To make it easier to create custom enricher extensions, you can use our boilerplate enricher: *eclecticiq-extension-example*.

It is a sample enricher that augments entities with social URI observables polled from Twitter and/or Facebook. Use it as a scaffold you can rework and customize into the desired enricher extension.

- Download, clone or copy the **eclecticiq-extension-example** (<https://github.com/eclecticiq/platform-extensions/tree/master/eclecticiq-extension-example>) extension, save it locally, and decompress it, if necessary.
- Rename the directories as needed.
- In the root directory, open **setup.py** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/setup.py>):

```
from setuptools import setup, find_packages

setup(
    name='eclecticiq-extension-example',
    version="1.0",
    description="Example extension for EclecticIQ Platform",

    # Look for packages to build the extension
    packages=find_packages(),

    # List dependencies the extension requires
    install_requires=[
        'eiq-platform'
    ],

    # Look for and include data files, if applicable
    include_package_data=True,

    # Set an entry point for the extension
    # so that it can initialize
    entry_points={
        'eiq.extensions': [
            'example = eiq.extensions.example:prepare_extension'
        ],
    }
)
```

Edit the setup file

These are the **setup.py** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/setup.py>) parts you can, and should, edit as applicable:

name

In the extension name, change `example` to a more meaningful name for your enricher extension, but leave the `eclecticiq-extension-` prefix as is.

Example:

```
name='eclecticiq-extension-fraud-ip-observables',
```

version

Change the `version` number as appropriate.

Make sure you implement versioning for your enrichers.

For example, increment the version number after adding or removing arguments/parameters.

The platform checks `version`, and it won't load the enricher if it detects parameter changes without a corresponding version number increment.

Example:

```
version="1.1",
```

description

Change the `description` value, so that it provides basic details about the enricher extension.

Example:

```
description="Custom extension to retrieve and save fraudulent IPs as observables",
```

Include dependencies

Add to the `install_requires` list the Python libraries and modules your enricher extension needs to access for it to work as expected.

`eiq-platform` is a mandatory dependency, it needs to always be included in the list.

Other Python libraries and modules you need to import and include in this list vary, depending on the specific enricher extension you are building.

Example:

```
install_requires=[
    'eiq-platform',
    'requests',
    'cabby',
    'furl'
],
```

Set the entry point

`eiq.extensions` is the designated entry point referring to the extension definitions.

The platform needs this pointer to recognize, load, and register extensions. Do not remove it.

Change `example` to a more meaningful name for your enricher extension, but leave the `eclecticiq.extensions.` prefix as is.

Example: `eclecticiq.extensions.fraud-ip-observables`

Example:

```
'eiq.extensions': [
    'fraud-ip-observables = eclecticiq.extensions.fraud-ip-observables:prepare_extension'
],
```


Define the enricher initialization

This part of the procedure customizes the fixtures you will need to run later to initialize the enricher, after enabling it.

`prepare_extension` defines how to initialize the enricher extension you are building.

In `prepare_extension` you specify what the extension should get ready before you execute it for the first time.

Change the `Extension` return values as applicable. Use the actual, correct names, descriptions, and values you define, set, and plan to use in your enricher extension.

Open ***init.py*** (https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/__init__.py).

Boilerplate:

```
from eiq.extension_api import Extension

from .enrichers import enrich_from_social_network
from .transformers import transform_malware_domainlist_csv
from .provider import oauth_http_download

def prepare_extension():
    return Extension(
        name=__name__,
        description='Example extension for EclecticIQ Platform',
        enrichers=[enrich_from_social_network],
        transformers=[transform_malware_domainlist_csv],
        transports=[oauth_http_download]
    )
```

Example:

```
# Import the 'Extension' class from the extension API
from eiq.extension_api import Extension

# Import your custom enricher
from .enrichers import enrich_fraud_ip_observables
# Import the data formats the enricher needs to handle
from .transformers import ${content_type}
# Import the data carrier the enricher uses to exchange data
from .provider import ${transport_type}

# Define how to initialize the custom enricher
def prepare_extension():
    return Extension(
        name=__name__,
        description='Custom extension to retrieve and save fraudulent IPs as observables',
        enrichers=[enrich_fraud_ip_observables],
        transformers=[${content_type}],
        transports=[${transport_type}]
    )
```

Start by importing the platform class defining extensions, as well as your custom extension, so that we can initialize it. You import the standard extension class for the platform through the `extension_api`. This is the dedicated API to use when developing custom extensions.

Import the enricher from ***enrichers.py*** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/enrichers.py>). This module defines, among others, enricher extension instance and type, as well as any functions containing the logic driving the enricher behavior, and the UI schema, if applicable.

Import one or more content types from ***transformers.py*** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/transformers.py>), so that the enricher can recognize and handle the expected data formats.

This module defines the available content types for the data the enricher extension handles: this is where you import the allowed data formats for the enricher from.

Import a transport type from ***provider.py*** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/provider.py>), so that the enricher can send and receive the expected data.

This module defines the available transport types for the data the enricher extension handles.

Change the `description`, `enrichers`, `transformers`, and `transports` metadata values `Extension` returns, so that they reflect the actual name, description, enricher type, data format, and data carrier to use in the enricher extension.

Build the enricher

Enricher extension logic and any necessary functions that drive the enricher behavior live in the ***enrichers.py*** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/enrichers.py>) module.

Boilerplate:

```
import json

import requests
from marshmallow import Schema, fields

# Import the enricher extension dependencies
from eiq.extensions_api import (
    EnrichmentResult,
    enricher_instance,
    enricher_type,
)

# Define the Marshmallow schema for the enricher params
# The UI schema is validated against the params defined here
class SocialURIEnricherSchema(Schema):
    check_twitter = fields.Boolean(required=True)
    check_facebook = fields.Boolean(required=True)

# Define the UI schema
ui_form_schema = [
    {
        "label": "Check Twitter",
        "name": "check_twitter",
        "required": True,
        "type": "checkbox",
        "format": "bool",
        "hint": "Should the enricher query Twitter"
    },
    {
        "label": "Check Facebook",
        "name": "check_facebook",
        "required": True,
        "type": "checkbox",
        "format": "bool",
        "hint": "Should the enricher query Facebook"
    }
]
```

```

        "name": "check_facebook",
        "required": True,
        "type": "checkbox",
        "format": "bool",
        "hint": "Should the enricher query Facebook"
    },
]

@enricher_instance(
    # Increment the version number if defaults change after
    # deploying the enricher.
    # For example, if it gets new/additional parameters
    version=1,

    # Enricher name and description displayed to users
    name='Example Social URI Enricher',
    description=('Query for registered Twitter/Facebook '
                'accounts with provided handle/name'),

    # Boolean switch to enable or disable the enricher by default
    is_active=True,

    # If the enricher creates entities, this is the default reliability
    # value assigned to them
    source_reliability='C',

    # Additional enricher parameters.
    # They must match the name and type configured in
    # 'ui_form_schema' and in the Marshmallow schema
    check_twitter=True,
    check_facebook=True,
)

@enricher_type(
    # Define the observable types the enricher supports
    input_extract_types=['handle', 'name', 'person'],

    # Assign a UI schema to the enricher editor page in the web UI
    parameter_ui_form_schema=ui_form_schema,

    # Define the validation and deserialization UI schema
    parameter_serialization_schema=SocialURIEnricherSchema,

    # URL templates link users to the enricher source
    source_urls={
        'handle': 'https://twitter.com/${input}',
        'name': 'https://twitter.com/${input}',
        'person': 'https://twitter.com/${input}',
    },
)

# Define the logic driving the enricher behavior
def enrich_from_social_network(
    # Request arguments
    extract_type,
    extract_value,
    check_twitter,
    check_facebook):
    # The response should return:
    # - A list with matching observables
    # - A list with the raw responses, for reference
    enrichment_extracts = []
    raw_responses = [] # Keep raw response headers for user reference

    if check_twitter:
        # Send an HTTP HEAD request to Twitter

```

```
# to see if it takes the handle
twitter_url = 'https://twitter.com/{}'.format(extract_value)
response = requests.head(twitter_url)

if response.ok:
    # Return and append the raw response(s)
    raw_responses.append(dict(response.headers))
    # Return and append supported observable types and values
    enrichment_extracts.append({
        'kind': 'uri',
        'value': twitter_url
    })

if check_facebook:
    # Send an HTTP HEAD request to Facebook
    # to see if it takes the handle
    facebook_url = 'https://facebook.com/{}'.format(extract_value)
    response = requests.head(facebook_url)

    if response.ok:
        raw_responses.append(dict(response.headers))
        enrichment_extracts.append({
            'kind': 'uri',
            'value': facebook_url
        })

return EnrichmentResult(
    # Return the raw responses as UTF-8 JSON
    raw_data=json.dumps(raw_responses).encode('utf-8'),
    # Return a key/value pair JSON list with the retrieved observables
    extracts=enrichment_extracts,
    # Return a list with entities, if applicable
    entities=[])
```

Let's break it down.

Import dependencies

Make sure you include the necessary Python libraries and modules in **enrichers.py**

(<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/enrichers.py>), so that the custom enricher extension can access the functionality required to work as expected.

The Python libraries and modules you may need to make available to your custom extension vary, depending on the extension design, scope, and purpose.

For example:

Dependency	Description
<code>import requests</code>	Adds handy automation to HTTP requests (http://docs.python-requests.org/en/master/).
<code>from marshmallow import Schema, fields</code>	Marshmallow schemas validate UI schemas and form input.

Dependency	Description
<code>from eiq.extensions_api import (EnrichmentResult, enricher_instance, enricher_type,)</code>	The foundations to build your enricher on.

```
import json

import requests
from marshmallow import Schema, fields

# Import the enricher extension dependencies
from eiq.extensions_api import (
    EnrichmentResult,
    enricher_instance,
    enricher_type,
)
```

Set the UI schema definition

If your enricher extension features a UI, you need to include a UI JSON schema, which you need to validate. The schema definition to validate UI schema and form input is based on a Marshmallow schema definition.

The **Marshmallow schema** (<https://marshmallow.readthedocs.io/>) defines the behavior of the controls and the components on the UI form, and the `ui_form_schema` JSON schema needs to match it to pass validation.

First, import the following classes from Marshmallow:

```
from marshmallow import Schema, fields
```

Then, set the Marshmallow schema definition to validate the `ui_form_schema` JSON schema against. You can customize the Marshmallow schema definition as needed.

Example:

```
# Define the Marshmallow schema for the enricher params
# The UI schema is validated against the params defined here
class SocialURIEnricherSchema(Schema):
    check_twitter = fields.Boolean(required=True)
    check_facebook = fields.Boolean(required=True)
```

Lastly, include `parameter_serialization_schema` in the `enricher_type` decorator in **`enrichers.py`** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/enrichers.py>), and set it so that it points to the appropriate schema definition name for the UI schema validation.

Example:

```
# Define the validation and deserialization UI schema
parameter_serialization_schema=SocialURIEnricherSchema,
```

Define the UI schema

If your enricher extension requires a UI frontend where users can make selections and set specific options, you need to include a UI schema in JSON format.

Each JSON field in the schema defines a UI component to implement in the extension. For example, an input field, or a checkbox.

Include the UI schema as a JSON array inside **enrichers.py** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/enrichers.py>). You can customize the UI schema as needed.

Example:

```
# Definition of the UI form rendered in the platform UI
# Data configuration > Enrichers > ${enricher_name} > Edit enricher task
# 'ui_form_schema' is the UI schema name; do not change it.
ui_form_schema = [
    {
        "label": "Check Twitter",
        "name": "check_twitter",
        "required": True,
        "type": "checkbox",
        "format": "bool",
        "hint": "Should the enricher query Twitter"
    },
    {
        "label": "Check Facebook",
        "name": "check_facebook",
        "required": True,
        "type": "checkbox",
        "format": "bool",
        "hint": "Should the enricher query Facebook"
    },
]
```

Then, include `parameter_ui_form_schema` in the `enricher_type` decorator in **enrichers.py** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/enrichers.py>), and set it so that it points to `ui_form_schema`.

Example:

```
# Assign the UI schema to the enricher editor page in the web UI
parameter_ui_form_schema=ui_form_schema,
```

You can define any UI schema that satisfies your requirements, provided it complies with the following guidelines:

- The UI schema format must be valid JSON.
- A UI schema for a form is a user-defined list of fields.

- Define each field with key/value pairs.
- Each key/value pair describes an attribute of the field.

Field attributes

You are free to define the naming convention and the terminology for the field names. However, field attributes are constrained and predefined. Each field takes at least two or more attributes.

`name` and `type` are required attributes, and you always need to include them in a field description. All other attributes are optional.

`name`

The name identifying the field in the JSON object.

This name is usually not displayed to users. It is included in the JSON object containing the field, the UI schema, and the extension schema that is returned when sending an API request to the `/private/extensions/` endpoint.

Example: *includeWhois*

`label`

The name of the field as displayed as a label on the resulting object in the UI form.

Example: *Include whois information*

`type`

It defines the type of field, that is, the object it represents on the UI form:

- `text`: a one-row text input field.
It can take the following sub-attributes:
 - `format`: it defines and restricts the allowed input format for the field, which needs to be in the specified format value.
Allowed values:
 - `datetime`
 - `host`
 - `url`
 - `email`
 - `regex`
 - `path`
 - `text`
 - `int`
 - `float`
 - `bool`
- `textarea`: a multiple row alphanumeric text input field.
- `password`: an alphanumeric input field that accepts a user password.
- `select`: a list with multiple options. Users can select one or more options.
It can take the following sub-attributes:
 - `options`: a JSON array with key/value pairs. Each key/value pair defines one option.
Format: `[{"name": "...", "value": "..."}, ...]`
 - `multiple`: Boolean, either `true` or `false`. It defines whether users are allowed to make multiple selections.

- **radio**: a control element that allows users to select one option in a set of options.
It can take the following sub-attributes:
 - **options**: a JSON array with key/value pairs. Each key/value pair defines one option.
Format: [{"name": "...", "value": "..."}, ...]
- **checkbox**: a control element that allows users to select/deselect, enable/disable an item or a feature.
It can take the following sub-attributes:
 - **options**: a JSON array with key/value pairs. Each key/value pair defines one option.
Format: [{"name": "...", "value": "..."}, ...]
 - If you do not include the **options** sub-attribute, the **checkbox** type defaults to a single component accepting Boolean values, either `true` or `false`.
- **extra**: include this type if you want to include in your UI form any additional free-form parameters, for example HTTP headers.
It can take the following sub-attributes:
 - **names**: a JSON array holding the name values of the extra free-form parameters. For example, the specific HTTP header names you want to add.
Format: ["name1", "name2", ...]
 - **allow_new**: Boolean, either `true` or `false`. It allows/denies adding new keys to the extra parameter list.

required

Boolean, either `true` or `false`.

It flags the field as either mandatory, that is, users must specify a value for the field, or optional.

default

Any value you specify for this attribute corresponds to the default value the field is pre-populated with (autofill).

hint

A tooltip text to give a short explanation of the field and the action the user should carry out.

Example: *Enter a numeric value between 1 and 10.*

when

It defines a conditional flow to show or hide the component when the specified criteria are met or not met.

Format: {"component_x": "value_y"}, that is, when `component_x` is set to `value_y`, the component the fields belongs to is displayed on the UI.

Define the enricher behavior

The foundations of your enricher extension are the `enricher_instance` and `enricher_type` decorators.

enricher_instance

This decorator defines the default values the enricher takes when it is instantiated in the platform.

The following arguments are mandatory:

- **version**: make sure you implement versioning for your enrichers.
For example, increment the version number after adding or removing arguments/parameters.
The platform checks `version`, and it won't load the enricher if it detects parameter changes without a corresponding version number increment.
- **name**: a human-readable, user-friendly name that helps user understand what the enricher does.
The `name` value is displayed on the UI.

- **description:** a human-readable, user-friendly, short free-text description that helps user understand what the enricher does.
The `description` value is displayed on the UI.
- **is_active:** the `True` or `False` Boolean value controls the state of the enricher **Enabled** checkbox on the UI: either selected or deselected, respectively.

The following arguments are optional:

- **source_reliability:** if your enricher extension is designed to create entities as an output, this parameter sets a default reliability value for the entity data source.
- Any custom parameters defining the Marshmallow schema definition to validate the UI schema.

Example:

```
@enricher_instance(
    # Bounce the version number if you
    # add, change or remove parameters
    version=1,

    name='Example Social URI Enricher',
    description=('Query for registered Twitter/Facebook '
                'accounts with provided handle/name'),

    is_active=True,
    # Optional, if your enricher returns entities
    source_reliability='C',

    # Custom Marshmallow UI schema definition values
    # to validate the enricher UI, if applicable
    check_twitter=True,
    check_facebook=True,
)
```

enricher_type

This decorator defines the enricher extension behavior. It contains the specific logic and the functionality powering your enricher.

The following arguments are mandatory:

- **input_extract_types:** a list of observable types.
The observable types in this list are the input data the enricher takes.
When it runs, the enricher searches for enrichment data to augment the observable types in the list.
- **parameter_ui_form_schema:** it needs to refer to `ui_form_schema`. Do not change the parameter name.
- **parameter_serialization_schema:** it needs to refer to the Marshmallow schema definition. Do not change the parameter name.
- **source_urls:** a key/value pair dictionary that produces clickable links to external data sources related to the retrieved observables. Do not change the parameter name.
 - The key name needs to be an observable type included in `input_extract_types`.
 - The value is a URL with the following format: `https://${data_source_url}/${input}`.
The `${input}` URL variable takes the `extract_value` value.

The resulting URL includes the original query as query URL params.

The resulting clickable link points to the original web site the enrichment data was obtained from.

Example:

```
@enricher_type(
    input_extract_types=['handle', 'name', 'person'],

    parameter_ui_form_schema=ui_form_schema,
    parameter_serialization_schema=SocialURIEnricherSchema,

    source_urls={
        'handle': 'https://twitter.com/${input}',
        'name': 'https://twitter.com/${input}',
        'person': 'https://twitter.com/${input}',
    },
)
```

def magic

Now it's time for the magic to happen. Define a workhorse function to do the grunt work.

The function you pass with the decorator should contain the necessary logic to search for and retrieve the desired input data, any conditional flow and error handling, and to output the input data as valid observables for the platform.

The following arguments are mandatory:

- `extract_type`: include this parameter, so that your function can return observable types matching `input_extract_types`.
- `extract_value`: include this parameter, so that your function can return values associated with the observable types defined in `input_extract_types`.
- Any custom-defined Marshmallow schema definition variables, if your enricher features a UI component.

The function should also create two variables to hold the returned data:

- `enrichment_extracts`: an empty list that will be populated with any matching observables included in the response.
- `raw_responses`: an empty list that will be populated with the raw responses, for reference.

Example:

```
def my_custom_enricher(
    # Mandatory arguments:
    extract_type,
    extract_value,
    ...,
    # Pass as arguments also the Marshmallow schema definition vars:
    check_twitter,
    check_facebook,
    ...
):

    # The function should return 2 lists:
    # - Any retrieved enrichment observables,
    # - The raw responses, for reference
    enrichment_extracts = []
    raw_responses = [] # Keep raw response headers for user reference

    ...
    # Your black magic happens here
    ...
```

return EnrichmentResult

Complete the function by defining the objects it should return in the response.

The `EnrichmentResult` class helps store and handle output data. It returns the following output:

- Raw response, that is, enrichment raw data (as UTF-8 JSON),
- Observables (as a list),
- Entities (as a list of entity IDs).

When building an enricher extension, it is a good idea to always use these data types to return, even when no data may be returned for observables or entities. The raw data response should always be included in the return arguments.

Example:

```
return EnrichmentResult(
    raw_data=json.dumps(raw_responses).encode('utf-8'),
    extracts=enrichment_extracts,
    entities=[])
```

Define transport and content handlers

Your enricher extension needs to include two more components:

- **provider.py** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/provider.py>): defines a mechanism to transport data. Typically, a provider implements a data transport protocol, such as HTTP, FTP, TAXII, and so on. Enrichers require a provider that fetches data from a source, and then passes it on to the platform for further processing.
- **transformers.py** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/transformers.py>): defines one or more mechanisms to convert retrieved raw data to the JSON format the platform can accept.

Define the data provider

To define the data provider for the enricher, you can use the **provider.py** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/provider.py>) as a scaffold you can rework and customize into the desired data transport provider for your enricher extension.

- Begin by importing the necessary Python modules and dependencies to handle the desired source data content type. In the example, the transport type is HTTP using OAuth authentication.
- `extension_api` is the dedicated API to use when developing custom extensions. From `extension_api`, import the following functions:
 - `options`: the `provider_options` passes it as an argument to control and validate UI options. Marshmallow uses it to set and validate dynamic form options in the UI.
 - `provider`: decorator function that marks a function as a provider type.
 - `provider_options`: decorator function to define a set of configuration options for a data provider.

```
from requests_oauthlib import OAuth1Session

from eiq.extensions_api import options, provider, provider_options
```

- The `provider` decorator flags a function as a data transport type/data provider in the platform. The decorator takes the following arguments:
 - `name`: an alphanumeric string holding the internal name identifying the data provider.
 - `title`: an alphanumeric string displayed in the UI. It corresponds to the transport type name for an enricher or an incoming feed.
 - `description`: an alphanumeric string displayed in the UI. It corresponds to a short description about the data provider/transport type purpose.

```
# The ``provider`` decorator will mark a certain function as a type
# of provider, along with some metadata primarily to inform users.
@provider(
    name='eiq.providers.oauth_http_download',
    title='OAuth HTTP Download',
    description=('Download content the configured URL using OAuth 1 '
                'authentication')
)
```

- The `provider_options` decorator enables you to define any custom or additional data provider settings that platform users can select and configure in the UI through the enricher form.

The decorator can take any custom argument key names, depending on how the source data provider interface works. Argument keys can take the following values:

- `options.boolean`: it works like a standard Python `bool` object; it takes either `True` or `False`. On the UI it displays a selection checkbox form element.
- `options.datetime`: it works like a standard Python `datetime.datetime` object. On the UI it displays a date-time calendar picker form element.
- `options.integer`: it works like a standard Python `int` object. On the UI it displays a text input form element.
- `options.password`: it works like a standard Python `str` object. On the UI it displays a password input form element.
- `options.string`: it works like a standard Python `str` object. On the UI it displays a text input form element.
- `options.url`: it works like a standard Python `str` object. On the UI it displays a text input form element that validates the input as a valid URL.

Each of these parameters can take additional *kwargs* to include extra information displayed to users on the UI:

- `label`: an alphanumeric string displayed in the UI. It corresponds to a UI option caption.
- `hint`: an alphanumeric string displayed in the UI. It corresponds to a pop-up tooltip triggered by a mouseover event.
- `required`: flags a UI option as required or optional. It takes either `True` or `False`, respectively.
- `default`: depending on the option type, it pre-populates the UI element with a default value.
- `validate`: it takes a function to validate user input with. Design the validation function so that it takes *one* value, and it returns `False` if the value is not valid.

```
# With ``provider_options`` additional configuration for a provider
# can be specified. These options will show up to users in the UI in
# an HTML form.
@provider_options(
    url=options.url(label="URL", hint="URL to download content from"),
    client_key=options.string(label="Client key"),
    client_secret=options.password(label="Client secret"),
    resource_owner_key=options.string(label="Resource owner key"),
    resource_owner_secret=options.password(label="Resource owner secret"),
)
```

Time for more magic: define a data provider to do the grunt work. The function you pass with the decorator should contain the necessary logic to handle data transport for the designated data format, and it should return/submit the data for further processing in the platform.

The function logic varies, based on your specific needs, and on the source data provider transport and content types. Make sure the function can `ctx.submit` or `return` the data for further processing, depending on how you design it.

The data provider function takes the following arguments:

- `ctx`: this must always be the first argument.
- Any arguments specified in `provider_options`.

```
# The provider function must accept a ``ctx`` object as its first
# argument. Other optional arguments are:
# :param content_type: The name of the related content_type for the incoming
#                       feed this function runs for.
# :param log: A logger instance
def oauth_http_download(
    ctx,
    url,
    client_key,
    client_secret,
    resource_owner_key,
    resource_owner_secret,
    log):

    oauth1_session = OAuth1Session(
        client_key,
        client_secret=client_secret,
        resource_owner_key=resource_owner_key,
        resource_owner_secret=resource_owner_secret)

    rs = oauth1_session.get(url)
    rs.raise_for_status()
    log.info("Fetched {} bytes of data".format(len(rs.content)))
    ctx.submit(rs.content)
```

Define the data transformer

To define the data format conversion mechanism for the enricher, you can use the ***transformers.py*** (<https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/eclecticiq/extensions/example/transformers.py>) as a scaffold you can rework and customize into the desired input-data-format-to-JSON converter for your enricher extension.

- Begin by importing the necessary Python modules and dependencies to handle the desired source data content type. In the example, the input data is in `.csv` format.

```
import csv
import io
```

- `extension_api` is the dedicated API to use when developing custom extensions.
From `extension_api`, import the following functions:
 - `declare_content_type`: it enables you to set the input data format your enricher expects to retrieve.
 - `transformer`: it enables you to configure the mechanism to convert the expected data input format to platform-friendly JSON.
It returns a dictionary containing `"type": "package"` similar to a STIX package, or a dictionary containing `"type": "entities"`, and `"entities": [<dicts with valid entity data>]`.

```
from eiq.extension_api import (
    declare_content_type,
    transformer,
)
```

- Define the input data format your enricher expects to retrieve by assigning it `declare_content_type` as a value.
In the example, the input data is in `.csv` format.
Use `declare_content_type` to define the data format by setting the following parameters:
 - `id`: the identifier value is a **URN** (<https://tools.ietf.org/html/rfc2141>).
Format: `urn:${namespace_identifier}:${namespace_specific_string}:${version}`
Example: `urn:malwaredomainlist:csv:1.0`
 - `name`: an alphanumeric string displayed in the UI. It corresponds to the enricher name.
 - `description`: an alphanumeric string displayed in the UI. It corresponds to a short description about the enricher purpose.
 - `file_extension`: sets the expected file extension type for the input data.
It needs to correspond to the MIME type.
Example: `csv`
 - `mime_type`: sets the expected **MIME type** (<https://www.iana.org/assignments/media-types/media-types.xhtml>) for the input data.
It needs to correspond to the file extension.
Example: `text/csv`

```
malware_domainlist_csv = declare_content_type(
    id="urn:malwaredomainlist:csv:1.0",
    name="Malware Domain List CSV",
    description="",
    file_extension="csv",
    mime_type="text/csv",
)
```

- The `transformer` decorator passes the function you define to handle data conversion from the configured input data format to platform-friendly JSON.
It takes an `incoming_content_types` list type argument, whose values are the defined input data format names.

Your custom function should always pass `blob` as an argument.

The function logic varies, based on your specific needs, and on the source data provider transport and content types. The generic steps you should consider when building such a function are:

- Set the expected data format and data encoding.
- Define the logic to iterate through the input data, and to extract the desired data.
- Create a variable to act as a container to hold the input raw data in a list.
- Return a JSON dictionary with the desired data.
- You may want to delegate entity or observable object creation to a separate function. In the example, it is the `make_entity` function.
 - Since the input data format in the example is `.csv`, the function takes `row` as a parameter to extract the desired data, which is then stored in two JSON objects:
 - `data`: a JSON array containing the extracted entity or observable object data.
 - `meta`: a JSON array containing the extracted entity or observable object metadata.
 - Return the `data` and `meta` JSON objects.
- The function you pass with the `transformer` decorator uses the entity or observable object creation function to return a JSON array with the extracted (linked) entities.

```
@transformer(incoming_content_types=[malware_domainlist_csv])
def transform_malware_domainlist_csv(blob):
    # The blob object is of type bytes.
    # The expected encoding from malwaredomainlist.com
    # is utf-8
    csv_data = blob.decode('utf-8')

    # DictReader expects to iterate over lines of text, so we wrap the data:
    csv_data = io.StringIO(csv_data)
    csv_reader = csv.DictReader(
        csv_data,
        fieldnames=[
            'date_utc',
            'domain',
            'ip',
            'reverse_lookup',
            'description',
            'registrant',
            'asn'
        ]
    )

    # List of entities generated from each extracted csv row
    entities = [make_entity(row) for row in csv_reader]

    # Return (linked) entities
    return {
        'linked-entities': {
            'entities': entities,
        },
    }

def make_entity(row):
    data = {
        'type': 'indicator'
```

```

        type = indicator,
        'title': row['domain'] + ' MalwareDomainList',
        'description': (
            'domain:          {}\n'
            'ip:              {}\n'
            'reverse lookup: {}\n'
            'description:     {}\n'
            'registrant:      {}\n'
            'asn:              {}'.format(
                row['domain'],
                row['ip'],
                row['reverse_lookup'],
                row['description'],
                row['registrant'],
                row['asn'])),
        'timestamp': row['date_utc'],
        'producer': {
            'description': 'Malware Domain List',
            'type': 'information-source',
        },
    },
]

meta = {
    'manual_extracts': [
        {'kind': 'uri', 'value': row['domain'], 'level': 1},
        {'kind': 'ipv4', 'value': row['ip'], 'level': 1},
        {'kind': 'domain', 'value': row['reverse_lookup'], 'level': 1},
        {'kind': 'registrar', 'value': row['registrant'], 'level': 1},
        {'kind': 'asn', 'value': row['asn'], 'level': 1},
    ],
}

return {'data': data, 'meta': meta}

```

Package and deploy the enricher

Create a Python package for the extension you just built, and then use `pip install` to install it on the target system where the platform is running.

- Pack the extension to create a source distribution by running the following command(s):

```
$ python setup.py sdist
```

- Copy the packaged extension to the target location where you want to deploy it.
- Activate a Python virtual environment:

```
$ source /opt/eclecticiq/platform/api/bin/activate
```

- In the virtual environment, install the extension by running the following command(s):

```
$ pip install /tmp/${your-custom-enricher-extension}.tar.gz
```

The `pip` example installs from `/tmp/` to avoid dealing with file access rights and permissions.

Restart processes

After completing the extension installation restart all *Supervisor* processes, so that all managed processes can configure the newly added extension in **Data configuration > Enrichers**.

- Reload Supervisor configurations and restart all Supervisor-managed processes by running the following command(s):

```
$ supervisorctl reload
```

Check that the enricher is registered

Make an API call with HTTPie

Verify that the extension is picked up and registered correctly.

To do so, save the following script to a *.sh* file, and then make it executable:

```
#!/bin/bash

#
# Helper for interacting with the platform API from the command line. This tool
# is a wrapper around httpie. It will take care of authentication and some
# other things. It takes at least three arguments:
#
# - host name
# - http method
# - relative url (without the /api/ part)
#
# Any additional arguments are propagated to httpie.
#
# Examples:
#
#   platform-api-http https://some.host/ GET /users/
#
#   platform-api-http localhost:8000 POST /some/endpoint/ @request-stored-in-a-file.json

set -e

readonly HTTPIE=http
readonly HTTPIE_ARGS="--check-status --verify=no"
readonly USERNAME=${platform_username}
readonly PASSWORD=${platform_password}

usage() {
    echo "Usage: $(basename $0) host method path [http-args]" > /dev/stderr
    exit 1
}

main () {
    local HOST="$1"
    local METHOD="$2"
    local API_PATH="$3"
    shift 3 || usage
    local TOKEN=$( ${HTTPIE} ${HTTPIE_ARGS} POST "${HOST}/api/auth" username=${USERNAME}
password=${PASSWORD} | jq --raw-output '.token')
    local URL="${HOST}/api${API_PATH}"
    ${HTTPIE} ${HTTPIE_ARGS} ${METHOD} ${URL} Authorization: '"Bearer ${TOKEN}"' "$@"
}

main "$@"
```

To make the script executable, run the following command(s):

```
$ chmod +x ~/${filename}.sh
```

The script takes the following input parameters:

Parameter	Description
<code>https://\${platform_host}/</code>	Required — The name of the host used to reach the API endpoint and to communicate with the API service.
<code>POST, GET, PUT, DELETE</code>	Required — A valid HTTP method (http://www.restapitutorial.com/lessons/httpmethods.html) to create, read, update, or delete a resource.

Parameter	Description
<code>/ \${API_endpoint} /</code>	<i>Required</i> — A relative URL pointing to the API endpoint that exposes the service you want to consume.
<code>?url=true&query=search-or-filter&params=4</code>	<i>Optional</i> — URL query parameters to send any additional search parameters and/or to filter the results returned in the response.



Besides appending URL query parameters, you can also send your request parameters as a JSON file.
Example:

```
$ platform-api-http https://${platform_host}/ get /entities/ @request-parameters.json
```

To make a **HTTPIe** (<https://httpie.org/>) call using the script, use the following format:

```
$ platform-api-http https://${host} ${method} ${api_path}
```

To check if the newly created enricher extension is correctly registered in the platform, make an API call to the `/extensions/` API endpoint:

```
$ platform-api-http https://${platform_host}.com get /extensions/
```

API call response

The call returns a JSON object containing all registered extensions.

Search for your enricher extension by name, description, or creation date.

If your enricher extension is included in the returned list, it is registered correctly.

Enable the enricher

- In the returned JSON object listing all registered extensions, search for your extension.
- In the extension JSON object, look for the following fields:
 - `id`: its value is a progressive integer that uniquely identifies the extension.
 - `is_active`: Boolean, either `True` or `False`. This flags the extension as either enabled or disabled, respectively.
- If `is_active` is set to `False`, the extension is currently disabled, and you need to enable it before you can use it.
To enable the extension, make the following API call:

```
$ platform-api-http https://${platform_host} put /extensions/${id_number} data='{ "data" : {  
  "is_active" : true } }'
```



When you pass a JSON object with entity data in the body of your API request, you always need to wrap it in a data wrapper: `{ "data" : { ... } }`

- Enabled extension names should *not* be included in the `DISABLED_EXTENSIONS` list in the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` configuration file.
Any extensions on this list are automatically disabled.
If an extension is on this list and you want to enable it, remove it from the list.

Initialize the enricher

The enricher extension is enabled, but not yet initialized. Platform enrichers though need to be initialized through fixtures before they become available.

Create and run the fixtures

- Log in to the system hosting the platform with either a user profile with admin rights, or with the `eclecticiq` user. You may need to grant the `eclecticiq` user admin privileges. If so, run the following command(s):
- Explicitly set the platform environment variable in the platform configuration file:

```
$ export EIQ_PLATFORM_SETTINGS=/opt/eclecticiq/etc/eclecticiq/platform_settings.py
```

- Activate a Python virtual environment:

```
$ source /opt/eclecticiq/platform/api/bin/activate
```

- Start a Python shell:

```
$ /opt/eclecticiq/platform/api/bin/eiq-platform shell
```

- In the Python shell, create the fixtures for the extensions by running the following command(s):

```
>>> from eclecticiq.extensions.boilerplate import create_fixtures
>>> create_fixtures()
```

Test the enricher

“Nah, my code doesn’t need testing.”

(anonymous, booted)

Run extension validation

The `eiq-platform extensions validate` command validates the state of all registered and enabled extensions.

- If the validation completes successfully, the command returns `Extensions look OK.`
- If the validation fails for one or more extensions, the command returns `Validation for {extension_name} failed:`, and it includes exception information for troubleshooting.

This command validates extensions based on the following criteria:

- The custom enricher extension name needs to match the `name` value defined in `setup.py`
- The extension name needs to be included in the platform `extension_registry` list, so that the platform can correctly recognize it and load it.
- The extension name should *not* be included in the `DISABLED_EXTENSIONS` list in the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` configuration file.
- The extension needs to support the configuration parameters of any enricher tasks associated with it.

To run `eiq-platform extension validate`, do the following:

- Explicitly set the platform environment variable in the platform configuration file;
- Run the command:

```
$ sudo -u eclecticiciq -i EIQ_PLATFORM_SETTINGS=/opt/eclecticiq/etc/eclecticiq/platform_settings.py /opt/eclecticiq/platform/api/bin/eiq-platform extensions validate
```

Error handling

If the validation detects issues that can prevent extensions from working correctly, it returns error messages with a short description of the problem.

Error message	Description
Validation for {extension_name} failed:	The enricher extension did not pass validation. Review the exception information included in the error message to start troubleshooting.
Extension '{extension_name}' is in the list of disabled extensions.	The extension name is included in the <code>DISABLED_EXTENSIONS</code> list in the <code>/opt/eclecticiq/etc/eclecticiq/platform_settings.py</code> configuration file. The extension enricher is disabled. Enable it.
Unknown extension '{extension_name}'	The extension name is not included in <code>DISABLED_EXTENSIONS</code> list, or in the <code>extension_registry</code> list. Start troubleshooting by checking packaging and deployment, and that the enricher extension is registered.

Error message	Description
Invalid parameters for enricher ({extension_name}). Errors: {returned_error_list}	One or more parameters are invalid. For example, they may be assigned a wrong format, or the enricher extension parameter schema is not validated correctly.
Task has parameters but enricher does not support any.	The enricher extension does not support one or more parameters defined in the enricher extension parameter schema.

Test the enricher with a test file

You can test your code programmatically by creating a test file that provides a valid sample request and a valid sample response for the enricher extension you built. The **`test_socialurienricher.py`** (https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/tests/test_socialurienricher.py) file provides a boilerplate to build your customized enricher extension test file.

The test file uses **HTTPretty** (<https://httpretty.readthedocs.io/en/latest/index.html>) to mock HTTP responses, and it makes REST API testing easy and transparent.

Check the **HTTPretty GitHub repository** (<https://github.com/gabrielfalcao/httpretty>) for more details and usage examples.

Example:

```
# Import the libraries, modules and classes you need to test your extension
import httpretty

from ${path.to.your.custom.enricher.extension} import (${CustomEnricherExtensionName})

@httpretty.activate
# This example uses dummy names and values.
# Replace them with the appropriate ones for your extension.
# This function mocks a HTTP 200 response.
def test_socialurienricher_found():

    # Mock the API endpoint and any additional URL params
    # Mock the HTTP status code the response should return
    httpretty.register_uri(
        httpretty.HEAD, 'https://twitter.com/jhonny', status=200)

    httpretty.register_uri(
        httpretty.HEAD, 'https://facebook.com/jhonny', status=200)

    # Mock the observable types and values the response should return
    result = enrich_from_social_network(
        extract_type='name',
        extract_value='jhonny',

        # Mock any UI-configurable settings
        check_twitter=True,
        check_facebook=True,
    )

    # Verify that the response returns
    # the expected amount of observables
    # generated from the retrieved data
    assert len(result.extracts) == 2

    assert all(e['kind'] == 'uri' for e in result.extracts)

    values = [e['value'] for e in result.extracts]

    assert 'https://facebook.com/jhonny' in values
    assert 'https://twitter.com/jhonny' in values

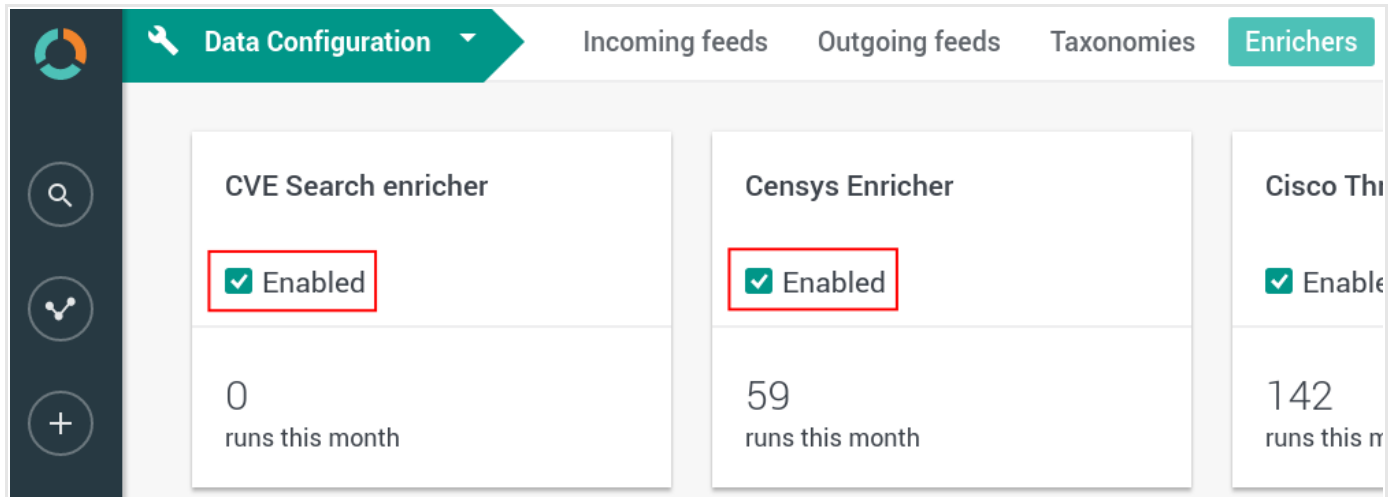
@httpretty.activate
# This function mocks a HTTP 404 response.
def test_socialurienricher_not_found():
    httpretty.register_uri(
        httpretty.HEAD, 'https://twitter.com/jhonny', status=404)
    httpretty.register_uri(
        httpretty.HEAD, 'https://facebook.com/jhonny', status=404)


    result = enrich_from_social_network(
        extract_type='name',
        extract_value='jhonny',
        check_twitter=True,
        check_facebook=True,
    )

    assert len(result.extracts) == 0
```

Test the enricher through the platform UI

- To check if your enricher extension is available in the platform UI, go to **Data configuration > Enrichers**.
- Your enricher extension should be displayed in the tiled overview, and the corresponding **Enabled** checkbox should be selected to notify that it is enabled.




- To test if your enricher extension works as expected, look for an entity with observables that your enricher extension supports.
- Trigger a manual enrichment:
 - On the entity detail pane, click **Observables**.
 - The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.
 - To manually enrich the entity observables:
- Click the  refresh icon to trigger a task run that polls all the enrichers configured for the entity.

Alternatively:

- From the **Actions** pop-up menu, select **Enrich > Enrich with all**.
- The platform polls all applicable enrichers for the entity, and it enriches all the entity observables with the retrieved data.
 - To poll a specific enricher:
- From the **Actions** pop-up menu, select **Enrich**, and then click the specific enricher whose task run you want to trigger.
- The platform polls the specified enricher for the entity, and it enriches all supported entity observables with the retrieved data.

×

Malicious files detected

 Ingested: 06.10.2017 9:20 Incoming feed: TAXII Stand Samples

TLP Not Set

OVERVIEW




OBSERVABLES





NEIGHBORHOOD

JSON

VERSIONS

HISTORY

<input type="checkbox"/>	Type	Value	Relation	Sighted	Conn.	First seen	Maliciousness	
<input type="checkbox"/>	hash-sha256:	e3b0c44298fc1c149afb4c899...	Related +1		2	06.10.2017 9:20	<div></div>	
<input type="checkbox"/>	hash-sha256:	d7a8fbb307d7809469ca9abcb...	Related +1		1	06.10.2017 9:20	<div></div>	
<input type="checkbox"/>	file:	readme.doc.exe	Related +1		1	06.10.2017 9:20	<div></div>	

Edit

Delete

Add to dataset

Add to graph

Create task

Export

Download original

Enrich

Enrich with all (5)

Censys Enricher

CrowdStrike Enricher

FireEye





Flashpoint AggregINT Enricher

Flashpoint Thresher Enricher

If you do not see any new observables after polling your enricher extension, check if the enricher crashed, and start investigating possible causes for the malfunction.

- In the platform UI, go to **Data configuration > Enrichers**.
On the enricher catalog page you can see a tile overview of the configured enrichers for the platform.
- Look for your enricher extension. If a (!) icon is displayed, the enricher task failed to run correctly.

Data Configuration Incoming feeds Outgoing feeds Taxonomies **Enrichers** Rules

CVE Search enricher <input checked="" type="checkbox"/> Enabled 0 runs this month	Censys Enricher <input checked="" type="checkbox"/> Enabled 65 runs this month	Circl.lu IP's related to SSL Certific... <input checked="" type="checkbox"/> Enabled 0 runs this month
Domaintools Domain Reputation <input checked="" type="checkbox"/> Enabled 58 runs this month	Domaintools Hosted Domains <input checked="" type="checkbox"/> Enabled 11 runs this month	Domaintools Malicious Server Do...  <input type="checkbox"/> Enabled 18 runs this month
Elastic Sightings Enricher  <input checked="" type="checkbox"/> Enabled 217 runs this month	Farsight DNSDB  <input type="checkbox"/> Enabled 26 runs this month	FireEye  <input type="checkbox"/> Enabled 148 runs this month

- Click the enricher tile.
- On the enricher detail page, click (!) **Failure**.

The screenshot shows the 'Enrichers' configuration page in the Eclectic IQ interface. The left sidebar contains navigation icons for search, view, and add. The main content area is divided into sections for configuration, state, rules, and enrichments.

Section	Field	Value
Configuration	Description	ElasticSearch Sightings Enricher
	Enabled	Yes
	Task name	eiq.enrichers.elastic_sightings
	Cache validity	2592000 second(s)
	Rate limit	1000 per second
	Monthly execution cap	100000 execution(s)
	Current month count	217 execution(s)
Observable types	Observable types	ipv6 domain uri hash-sha512host ipv4 Show all (8)
	Parameters	Elastic search url: http://localhost:9200/test Username: test Password: **** Observable queries: Hash-sha256 "report-{value}"
State	State	❗ FAILURE
Enrichment rules	Enrichment rules	stresss EnricherTestRule
Enrichments	Enrichments	Last 7 days: Day: 2017-08-15 Count: 46 Day: 2017-08-16 Count: 4 Day: 2017-08-17 Count: 7 Day: 2017-08-21 Count: 1 Day: 2017-08-22 Count: 11

- An error dialog is displayed. The dialog title notifies the type of error, whereas the traceback area gives a detailed stack trace in reverse chronological order. The stack trace should give you at least some hints about the possible causes of the failure.

FAILURE

Run started

Last updated / finished

Total runtime *Unknown*

Time period of data request *Unknown , from to*

Error **Revoked**

Traceback

```
404 Client Error: Not Found
Traceback (most recent call last):
  File "/opt/eclecticiq/sources/platform-api/eiq/platform/taskrunner/base.py", line 138, in run
    result = self.work(run_parameters=run_parameters)
  File "/opt/eclecticiq/sources/platform-api/eiq/platform/enrichments.py", line 366, in work
    enrichment_result = self.enrich(input_extract)
  File "/opt/eclecticiq/platform/api/lib/python3.4/site-packages/eiq/extensions/sighting_enrichers/elasticsearch.py", line 122, in enrich
    extract, task.parameters)
  File "/opt/eclecticiq/platform/api/lib/python3.4/site-packages/eiq/extensions/sighting_enrichers/elasticsearch.py", line 151, in retrieve_sighting_info
    search_result = elastic_search(es_query, parameters)
  File "/opt/eclecticiq/platform/api/lib/python3.4/site-packages/eiq/extensions/sighting_enrichers/elasticsearch.py", line 179, in elastic_search
    rs.raise_for_status()
  File "/opt/eclecticiq/platform/api/lib/python3.4/site-packages/requests/models.py", line 851, in
    raise_for_status
    raise HTTPError(http_error_msg, response=self)
requests.exceptions.HTTPError: 404 Client Error: Not Found
```

stresss
EnricherTestRule

Disable extensions

To disable an enricher extension and prevent the platform from loading it, do the following:

- Open `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` in read-write mode.
- Browse to the `DISABLED_EXTENSIONS` parameter.
If it is not there, add it to the settings file.
This parameter is a list holding the `name` values of all disabled extensions, as defined in the `setup` file.
- Add your enricher extension `name` value to the list.
Example:

```
DISABLED_EXTENSIONS = [
    'eclecticiq-extension-fraud-ip-observables',
    ...,
]
```

- Save the settings file and exit.

Enrichers reference

Reference section with lookup information on the default enrichers, enricher inputs and outputs, and a list with the supported enrichment data types. Enrichers augment existing cyber threat intelligence value by adding contextual information.

Enrichers

Enrichers poll external data sources to provide additional context and detail to augment — hence, enrich — the intelligence value of entities and observables stored in the platform.

The platform ships with a broad selection of built-in, ready-to-use enrichers to obtain geolocation IP and whois details, DNS domain and malware information, as well as other relevant data to help analysts draw a sharper and more comprehensive picture of the cyber threat relationships and the cyber threat scenarios under investigation.

Enrichers automatically augment all the entities that accept the enricher's content type as an observable. In other words, the observable types an entity supports define the applicable enrichers an entity can use.

Default enrichers

Title	Excerpt
Censys enricher	The Censys enricher returns a wealth of information about IP addresses, from a network ASN to their geographic location, so that you can explore relationships between events, actors, and targets.
CIRCL IPs related to SSL certificate enricher	The CIRCL IPs related to SSL certificate enricher uses the CIRCL API to poll the CIRCL Passive SSL database and to obtain all IP addresses associated with the input SSL SHA-1 hash fingerprints.
CIRCL SSL Certificate Fetcher enricher	The CIRCL SSL Certificate Fetcher enricher uses the CIRCL API to poll the CIRCL Passive SSL database and to obtain parsed SSL certificates and all domain names associated with the input SSL SHA-1 hash fingerprints.
Cisco Threat Grid enricher	The Cisco Threat Grid enricher provides information on a range of cyber threat data like IP addresses, domains, registry keys, network streams, and hash files.
CrowdStrike Falcon Intelligence Indicator enricher	The CrowdStrike Falcon Intelligence Indicator enricher returns observables extracted from indicators to provide additional context to existing platform intelligence.
CVE Search enricher	The CVE Search enricher contacts CIRCL cve-search API to poll information about common software and hardware vulnerabilities, along with the corresponding exposures.
DomainTools Hosted Domains enricher	The Domaintools Hosted Domains enricher returns all domain names related to the specified input IP addresses.

Title	Excerpt
DomainTools Malicious Server Domains enricher	The DomainTools Malicious Server Domains enricher returns malicious domain names related to the same primary and/or secondary name servers, along with their risk scores to automatically flag server domains with an appropriate maliciousness confidence level.
DomainTools Parsed Whois enricher	The DomainTools Parsed Whois enricher returns malicious domain names related to the same primary and/or secondary name servers.
DomainTools Reputation enricher	The Domaintools Reputation enricher returns risk scores to assess the reputation of the specified input domain and host names.
DomainTools Reverse Whois enricher	The DomainTools Reverse Whois enricher returns malicious domain names related to the name, address, telephone number, email address or physical address of the registrant listed in current or historical whois records.
DomainTools Suspicious Domains enricher	The DomainTools Suspicious Domains enricher returns suspicious and potentially malicious domains related to the input IP addresses, along with their risk scores to automatically flag domains with an appropriate maliciousness confidence level.
Elasticsearch sightings enricher	The Elasticsearch sightings enricher creates sightings from matching results returned from a search in an external Elasticsearch instance.
Create custom enrichers	Implement custom extensions to integrate EclecticIQ Platform with external intel providers and data sources through incoming feeds and enrichers, as well as to publish platform intel downstream in your prevention and detection toolchain.
Farsight DNSDB enricher	The Farsight DNSDB enricher provides historical passive DNS information to relate domain names to the IP addresses they point to, or IPs pointing to different domains over time.
FireEye iSIGHT enricher	The FireEye iSIGHT enricher retrieves threat intelligence reports about threats and malware related to areas such as critical infrastructure, cyber crime and espionage, hacktivism, frauds, and vulnerability and exploitation.
Flashpoint AggregINT enricher	The Flashpoint AggregINT enricher provides information on a range of cyber threat data like actor information, email and IP addresses, domains, and hash files. It focuses on threat actors and criminal groups.
Flashpoint Blueprint enricher	The Flashpoint Blueprint enricher provides information on a range of cyber threat data like actor information, email and IP addresses, domains, and hash files. It focuses on threat actors and criminal groups.
Flashpoint Thresher enricher	The Flashpoint Thresher enricher provides information on a range of cyber threat data like actor information, email and IP addresses, domains, and hash files. It focuses on threat actors and criminal groups.
Fox-IT InTELL Portal enricher	The Fox-IT InTELL Portal enricher provides information from a range of sources, such as forums and sites that have registered potentially suspicious activity.
Intel 471 enricher	The Intel 471 enricher provides information on compromised IP addresses, domains, URLs, and emails, as well as first-hand information about threat actors and groups.
MaxMind GeoIP enricher	The MaxMind GeoIP enricher maps input IP addresses to their corresponding geographic locations by polling data from an offline MaxMind database.

Title	Excerpt
MISP API enricher	The MISP API enricher retrieves data from a source MISP instance. It returns enrichment information on indicators of compromise. Based on the enricher criteria and the retrieved content, ingested data is stored to the platform as enrichment observables, entities, and relationships.
NSFocus Intelligence enricher	The NSFocus enricher augments input observables with a wide range of contextual information such as geolocation, ASN, whois, registrar, domain name, file hash, and so on.
OpenDNS OpenResolve enricher	The Cisco OpenDNS OpenResolve enricher provides reverse-DNS lookup information.
PassiveTotal Passive DNS enricher	The PassiveTotal Passive DNS enricher provides historical information to enable cross-referencing IP addresses to the corresponding DNS domain names over time.
PassiveTotal IP/Domain enricher	The PassiveTotal IP/Domain enricher provides additional information for the queried IP address or domain name.
PassiveTotal Malware enricher	The PassiveTotal Malware enricher provides malware information related to the queried host or domain, such as malware hash and collection date.
PassiveTotal Whois enricher	The PassiveTotal Whois enricher provides information about individuals or entities associated with an IP address or a domain name, as well as geolocation details.
Proofpoint Email Threat enricher	The Proofpoint Email Threat enricher uses input data such as email subjects, domain names, hashes, and IP addresses to return information on email threats such as phishing, spoofing, and email malware. The ingested Proofpoint data is stored to the platform as email threat indicators.
PyDat enricher	The PyDat enricher provides whois, including historical whois, and passive DNS lookup information.
Recorded Future enricher	The Recorded Future enricher enables you to tap into the data stream generated by the Recorded Future Temporal Analytics Engine to retrieve search results potentially malicious IPs, domains, email addresses, and hashes related to the input observable types, along with their risk scores to automatically flag domains ...
RIPEstat GeoIP enricher	The RIPEstat GeoIP enricher provides geolocation information related to the queried IP addresses.
RIPEstat Whois enricher	The RIPEstat Whois enricher provides whois information related to the queried IP addresses.
Shodan enricher	The Shodan enricher uses input data such as country and city names, organization and personal names, ZIP codes, email addresses, and so on to return a list of matching IP addresses corresponding to your Internet-connected devices, along with location and user details.
Splunk sightings enricher	The Splunk sightings enricher searches the indexes in the specified Splunk instance. Matching data is extracted and saved to the platform as sightings.
SpyCloud Breach Data enricher	The SpyCloud Breach Data enricher uses input data such as IP addresses, domain names, email addresses and user access credentials to return incident and security breach information that is processed as incident entities, related observables, and enrichment observables.

Title	Excerpt
ThreatCrowd enricher	The ThreatCrowd enricher returns suspicious and potentially malicious domains, IP addresses, email addresses, file hashes, and antivirus detections, so that you can explore relationships between events, actors, and targets.
Unshorten-URL enricher	The Unshorten-URL polls the specified URL shortener services to return the resolved original URLs corresponding to the submitted shortened ones.
VirusTotal enricher	The VirusTotal enricher provides information on malware, domains (passive DNS) and IP addresses.

Enricher types

Enricher	API endpoint	Description
Elasticsearch sightings	<code>http://\${elasticsearch_instance_url}:9200/\${schema_resource}</code>	Searches an external Elasticsearch instance. Search criteria are processed to sightings.
Fox-IT InTELL Portal	<code>https://cybercrime-portal.fox-it.com/</code>	Based on Fox-IT InTELL, the portal provides a range of sources, such as for potentially suspicious activity.
Intel 471	<code>https://api.intel471.com/v1/</code>	Besides data on compromised Intel 471 focuses on providing information on actors and groups.
OpenDNS OpenResolve	<code>http://api.openresolve.com/{}/{}</code>	OpenResolve by OpenDNS offers a way to retrieve reverse-DNS lookup information.
PyDat	<code>http://\${pydat_instance_url}:8000/</code>	PyDat (https://github.com/mitre-internal/pydat) is a Python library that can work together with Elasticsearch (https://github.com/mitre-internal/pydat-with-elasticsearch) to provide passive DNS lookup information, organization, country, city, street, and more.
RIPEstat GeolIP	<code>https://stat.ripe.net/data/geoloc/{}</code>	Geolocation IP information from the RIPEstat API (https://stat.ripe.net) including longitude, country, and city.
RIPEstat Whois	<code>https://stat.ripe.net/data/whois/{}</code>	Whois information from the RIPEstat API (https://github.com/ripe/ripestat-api) including inet number, name, or telephone.
Cisco Threat Grid	<code>https://panacea.threatgrid.com/api/v3/</code>	Polls data from the Cisco Threat Grid range of cyber threat data like IP network streams, and hash files.

Enricher	API endpoint	Details
VirusTotal	https://www.virustotal.com/vtapi/v2/{}	Polls data from the VirusTotal A domains (passive DNS) and IP against 60+ antimalware products. It produces additional metadata information.
Flashpoint AggregINT	https://endlesstunnel.info/v3	Polls data from the Flashpoint A hosts, domains, IP addresses, and thematic datasets focusing on hacker groups, communities in conflict, and CBRN (https://en.wikipedia.org/wiki/Chemical_biological_radiological_nuclear). It produces enrichment observables like user name of the author of a post in a forum, UTC date and time of a post in a forum, (https://en.wikipedia.org/wiki/Unicode) (https://tools.ietf.org/html/rfc5200).
Flashpoint Blueprint	https://endlesstunnel.info/v3	Polls data from the Flashpoint A geolocation and IP ranges, as well as search thematic datasets focusing on supremacist groups, state actors, and CBRN (https://en.wikipedia.org/wiki/Chemical_biological_radiological_nuclear). It produces enrichment observables like city, latitude/longitude or IP address, a hit, user name uniquely matched.
Flashpoint Thresher	https://endlesstunnel.info/v3	Polls data from the Flashpoint A datasets focusing on hackers, threats, and CBRN (https://en.wikipedia.org/wiki/Chemical_biological_radiological_nuclear). It produces enrichment thresher data.
PassiveTotal Whois	https://api.passivetotal.org/v2	Polls data from the PassiveTotal (https://api.passivetotal.org/v2/getv2whoisquery). It provides associated with an IP address details. Analysts can retrieve related telephone, and email details. They can further queries to obtain, for example, with the same individual or the same domain.
PassiveTotal Passive DNS	https://api.passivetotal.org/v2	Polls data from the PassiveTotal (https://api.passivetotal.org/v2/getv2dnspassivequery). It provides cross-referencing IP addresses over time. Analysts can examine different IP addresses over time and retrieve more domain names than the original IP address.

Enricher	API endpoint	Description
PassiveTotal IP/Domain	https://api.passivetotal.org/v2	<p>Polls data from the PassiveTotal (https://api.passivetotal.org/v2/enrichmentquery). It processes the queried IP address or domain name, any sub-domains, internet service providers (ISP), and Autonomous System (ASN) (https://en.wikipedia.org/wiki/AS) as well as geolocation information to look for further connections throughout the investigation.</p>
PassiveTotal Malware	https://api.passivetotal.org/v2	<p>Polls data from the PassiveTotal (https://api.passivetotal.org/v2/enrichmentmalwarequery) to the queried host or domain, such as sha1, hash-sha256, hash-sha512. Malware entries are also tagged with <code>enrichment_extracts.meta.confidence</code> to the value you set under Data Configuration > Action > Mark as malicious or <code>enrichment_extracts.meta.confidence</code> to the value you set under Data Configuration > Malicious - Low Confidence > Malicious - Low.</p>
Splunk sightings	http://\${splunk_instance_url}:8089/	Based on the search queries defined in the configuration, it searches for matching data in the specific index extracted and saved to the platform.
DomainTools Hosted Domains	http://api.domaintools.com/v1/{}/host-domains	Enriches IPv4 observables by resolving IP addresses and therefore related to, the IP addresses.
DomainTools Reputation	http://api.domaintools.com/v1/reputation	Enriches domain and host name information to assess maliciousness based on defined threshold values.
DomainTools Suspicious Domains	https://api.domaintools.com/v1/{}/host-domains	Enriches IPv4 observables with IP addresses. It includes configuration options to set confidence levels to the process of identifying malicious IPs.
FireEye iSIGHT	https://api.isightpartners.com/search/{}	Enriches platform observables with threat intelligence from www.threatcrowd.org and other critical infrastructure, cyber crime, vulnerability and exploitation.
Recorded Future	https://api.recordedfuture.com/live/sc/entity/{}	The enricher returns additional IP addresses, and hashes related to the specified types, as well as maliciousness and retrieved risk scores.
Unshorten- URL	https://unshorten.me/s/{}	It takes shortened URL as an input and returns resolved original URLs, which can be used to discover relationships with other observables.

Enricher	API endpoint	Description
Farsight DNSDB	https://api.dnsdb.info/ {}	Historical passive DNS lookup e pointing to a specified IP address nameserver, domain names poi existing below a parent domain
ThreatCrowd	https://www.threatcrowd.org/ {}	Returns suspicious and potentia addresses, file hashes, and anti relationships between events, a
Censys	https://censys.io/api/v1/search/ipv4	Returns relevant contextual info types to augment their intelligen details, hashes, and ASN (https://en.wikipedia.org/ details. It makes it easier to disc actors, and targets.
DomainTools Malicious Server Domains	http://api.domaintools.com/v1/ {}/name-server-domains/	Enriches domain and host obse domain names related to the sa includes configurable threshold levels to the processed domains domains/hosts.
DomainTools Parsed Whois	http://api.domaintools.com/v1/ {}/whois/parsed	Enriches domains, hosts, and IF JSON output includes the most domain or IP range, as well as p registrar, contacts, and so on. It referencing data in a set of Whc
CrowdStrike Falcon Intelligence Indicator	https://intelapi.crowdstrike.com/indicator/v1/search/ {}	Enriches platform entities and o IP addresses, domain names, e
DomainTools Reverse Whois	http://api.domaintools.com/v1/reverse-whois/ {}	Enriches supported observable current or historical Whois recor actors.
CVE Search	https://cve.circl.lu/api/cve/	Enriches supported observable software and hardware vulnerat exposures. The enricher contac Center Luxembourg (CIRCL) cv details associated with the input common software and hardware corresponding exposures, is stc observables.
CIRCL IPs related to SSL certificate	https://www.circl.lu/v2pssl/cquery/ {}	The enricher contacts the Comp Luxembourg (CIRCL) API and t retrieve all observed IP address fingerprint *hash-sha1* observa holds historical data matching S

Enricher	API endpoint	Description
CIRCL SSL Certificate Fetcher	<code>https://www.circl.lu/v2pssl/cfetch/{}</code>	The enricher contacts the Comp Luxembourg (CIRCL) API and retrieve the parsed certificate ar with the input SSL fingerprint *h Passive SSL database holds his fingerprints with the correspond domain names, that is, the certi `subjectAltName`/*Subject Alter
Shodan	<code>https://api.shodan.io/shodan/</code>	Shodan matches input data suc company or personal names, to discovering which devices in yo to the Internet, where they are l
SpyCloud Breach Data	<code>https://api.spycloud.io/sp-v1/breach</code>	Enriches supported observable information about account takec description of the incident, targ of stolen records — for example credentials — as well as when t sale on the Internet. Each entry entry of a breached database c password, and email address) -
Proofpoint Email Threat	<code>https://api.emaildefense.proofpoint.com/v1</code>	Enriches supported observables threats such as phishing, spoofi email/fraudulent messages.
MISP API	<code>http://\${misp_instance_url}/</code>	Enriches supported observables information.
MaxMind GeoIP	<code>/absolute/path/to/GeoLite2-City.mmdb</code>	Enriches input IP addresses wit from an offline MaxMind GeoLit at city level.
NSFocus Intelligence	<code>https://nti.nsfocusglobal.com/api/v1/search/{}</code>	Enriches supported observables information such as related cou registrar and whois.

Enricher input

The overview shows the supported observable data types you can use as input for the enrichers. These are the value types the *enrichment_extracts.kind* search query field returns.

Enricher	Supported kinds (observable types)
Elasticsearch sightings	domain, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
Fox-IT InTELL Portal	domain, hash-md5, hash-sha1, hash-sha256, host, ipv4, ipv6, uri
Intel 471	actor-id, domain, email, hash-md5, hash-sha256, host, ipv4, ipv6, uri

Enricher	Supported kinds (observable types)
OpenDNS OpenResolve	domain, host, ipv4, ipv6
PyDat	domain, ipv4, ipv6
RIPEstat GeoIP	ipv4, ipv6
RIPEstat Whois	ipv4, ipv6
Cisco Threat Grid	domain, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri, winregistry
VirusTotal	domain, hash-md5, hash-sha1, hash-sha256, ipv4, ipv6, uri
Flashpoint AggregINT	actor-id, domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
Flashpoint Blueprint	actor-id, domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
Flashpoint Thresher	domain, file, hash-sha1, host, ipv4, uri
PassiveTotal Whois	domain, host, ipv4, ipv6
PassiveTotal Passive DNS	domain, host, ipv4, ipv6
PassiveTotal IP/Domain	domain, host, ipv4, ipv6
PassiveTotal Malware	domain, host
Splunk sightings	domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, uri
DomainTools Hosted Domains	ipv4
DomainTools Reputation	domain, host
DomainTools Suspicious Domains	ipv4
FireEye iSIGHT	asn, domain, email, email-subject, file, hash-md5, hash-sha1, hash-sha256, host, ipv4, ipv6, uri
Recorded Future	domain, hash-md5, hash-sha1, hash-sha256, hash-sha512, ipv4, ipv6
Unshorten-URL	uri
Farsight DNSDB	domain, host, ipv4, ipv6
ThreatCrowd	domain, email, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, malware
Censys	asn, city, company, country, country_code, geo-lat, geo-long, hash-md5, hash-sha1, hash-sha256, ipv4, postcode
DomainTools Malicious Server Domains	domain, host

Enricher	Supported kinds (observable types)
DomainTools Parsed Whois	domain, host, ipv4
CrowdStrike Falcon Intelligence Indicator	domain, email, email-subject, file, hash-md5, hash-sha1, hash-sha256, ipv4, ipv6, mutex, name, persona, port, uri
DomainTools Reverse Whois	address, asn, city, company, country, country-code, email, name, organization, person, postcode, street, telephone
CVE Search	cve
CIRCL IPs related to SSL certificate	hash-sha1
CIRCL SSL Certificate Fetcher	hash-sha1
Shodan	asn, city, company, country, country-code, domain, email, geo-lat, geo-long, hash-md5, hash-sha1, hash-sha256, hash-sha512, host, ipv4, ipv6, organization, person, port, postcode, uri
SpyCloud Breach Data	domain, email, handle, ipv4, ipv6
Proofpoint Email Threat	domain, email-subject, hash-md5, ipv4, uri
MISP API	actor-id, bank-account, card, country, domain, email, email-subject, file, hash-md5, hash-sha1, hash-sha256, hash-sha512, ipv4, ipv6, malware, name, netname, organization, person, port, registrar, telephone, uri, yara
MaxMind GeoIP	ipv4
NSFocus Intelligence	cve, domain, hash-md5, ipv4, uri

Enricher output

Enrichers automatically augment all the entities that accept the enricher's content type as an observable. In other words, the observable types an entity supports define the applicable enrichers an entity can use.

The overview describes the output each enricher generates. The resulting enrichment observables are associated with the entities they bear relationships to.

Enricher	Generated output
Elasticsearch sightings	Creates sightings from matching results returned from a search in an external Elasticsearch instance.
Fox-IT InTELL Portal	Enriches supported observable types with relevant contextual information from forums, chats, and IRC channels.
Intel 471	Enriches supported observable types with data focusing on threat actor information.

Enricher	Generated output
OpenDNS OpenResolve	Enriches supported observable types with reverse-DNS lookup information.
PyDat	Enriches supported observable types with whois data, current IP resolution and passive DNS information.
RIPEstat GeoIP	Enriches supported observable types with geolocation information related to IP addresses: coordinates, country, and city.
RIPEstat Whois	Enriches supported observable types with whois information related to IP addresses.
Cisco Threat Grid	Enriches supported observable types, as well as all found observables based on the enricher configuration, with information such as IP addresses, domains, host names, hashes, and Windows registry keys.
VirusTotal	Enriches supported observable types with maliciousness confidence level information.
Flashpoint AggregINT	Enriches supported observable types with information such as IP addresses, domains, host names, and hash files.
Flashpoint Blueprint	Enriches supported observable types with information such as IP addresses, domains, host names, and URLs.
Flashpoint Thresher	Enriches supported observable types with information such as IP addresses, domains, URLs, hashes, and files.
PassiveTotal Whois	Enriches supported observable types with whois (https://www.riskiq.com/products/learn-threat-research-and-analysis/) information.
PassiveTotal Passive DNS	Enriches supported observable types with passive DNS (https://www.riskiq.com/products/learn-threat-research-and-analysis/) information.
PassiveTotal IP/Domain	Enriches supported observable types with enrichment (https://passivetotal.readthedocs.io/en/latest/api.html#enrichment-request) information.
PassiveTotal Malware	Enriches supported observable types with malware enrichment (https://passivetotal.readthedocs.io/en/latest/api.html?highlight=malware#enrichment-request) information.
Splunk sightings	Creates sightings for matching input observables, based on the search result items retrieved in the specified Splunk instance.
DomainTools Hosted Domains	Enriches supported observable types with domain and host name information.
DomainTools Reputation	Enriches supported observable types with reputation information.
DomainTools Suspicious Domains	Enriches supported observable types with suspicious domain and host name information.
FireEye iSIGHT	Enriches supported observable types related to the matching input observable types.

Enricher	Generated output
Recorded Future	Enriches supported observable types with pattern matching search results produced by the Recorded Future Temporal Analytics Engine.
Unshorten-URL	Original URL corresponding to the submitted shortened one.
Farsight DNSDB	Enriches supported observable types with passive DNS lookup information such as the name of the domain or host name owner, or the IP address a domain or host name points to.
ThreatCrowd	Enriches supported observable types with suspicious and potentially malicious domains, IP addresses, email addresses, file hashes, and antivirus detections.
Censys	Enriches supported observable types by providing additional context such as geolocation, country and city information, as well as ASN (https://en.wikipedia.org/wiki/autonomous_system_(internet)) details.
DomainTools Malicious Server Domains	Enriches supported observable types with malicious domain names that are served from the same name server.
DomainTools Parsed Whois	Enriches supported observable types with structured Whois information.
CrowdStrike Falcon Intelligence Indicator	Enriches supported observable types with information extracted from indicators.
DomainTools Reverse Whois	Enriches supported observable types with reverse Whois information.
CVE Search	Enriches supported observable types with CVE details.
CIRCL IPs related to SSL certificate	Enriches SSL fingerprint hash observables with all associated IP addresses.
CIRCL SSL Certificate Fetcher	Enriches SSL fingerprint hash observables with the parsed certificate and all associated domain names.
Shodan	Enriches supported observable types with the following information, when available: country name, city name, ZIP code, longitude, latitude, organization name, host name, IP address, open ports and services related to input IP addresses.
SpyCloud Breach Data	Enriches supported observable types and incident entities with account takeover (ATO) and security breach details. Generates Cybox observables, related observables, and CIQ objects. Supported output observables: <i>city</i> , <i>country-code</i> , <i>domain</i> , <i>email</i> , <i>geo-lat</i> , <i>geo-long</i> , <i>handle</i> , <i>host</i> , <i>ipv4</i> , <i>ipv6</i> , <i>organization</i> , <i>person</i> , <i>postcode</i> , <i>telephone</i> .
Proofpoint Email Threat	Enriches supported observables and entities with information on email threats. Generates email threat indicators containing observables that represent the ingested data, as well as relationships between the enriched entities and the enrichment data. The output includes, when available, the brand a spoofing message uses to trick users, the message subject line, source IP address, source host name, and source email address, as well as any MD5 file hashes, if the malicious email contains attachments.
MISP API	Enriches supported observables and entities with information on indicators of compromise. Searches a source MISP instance for MISP events containing the values of the selected observable types. Generates enrichment observables, entities, and relationships based on the retrieved data.

Enricher	Generated output
MaxMind GeoIP	Geographic location information to detect the physical position of the input IP addresses.
NSFocus Intelligence	Details on geolocation, ASN, registrar, whois, running ports and protocols, domain names, and file hashes.

Enrichment data types

actor-id
address
asn
bank-account
card
card-owner
cce
city
company
country
country-code
cve
cwe
domain
email
email-subject
eui-64
file
forum-name
forum-room
forum-thread
fox-it-portal-uri
geo
geo-lat

geo-long
handle
hash-md5
hash-sha1
hash-sha256
hash-sha512
host
industry
inetnum
ipv4
ipv6
mac-48
malware
mutex
name
nationality
netname
organization
person
port
postcode
process
product
registrar
rule
snort
street
telephone
uri
uri-hash-sha256
winregistry

yara