EclecticIQ Platform integrations

Integrate with external tools in your cyber security ecosystem

Last generated: July 21, 2017

Table of contents

EclecticIQ Platform integrations

EclecticIQ Platform integrations with third-party products and systems to leverage external intel sources and improve collaboration.

Browse the table for the topics you want to look up.
You can also use the drop-down menu on the left-hand navigation sidebar to access the articles or to go to a different section.

Title	Excerpt
Cisco OpenDNS integration	Integrate EclecticIQ Platform with Cisco OpenDNS through OpenResolve by OpenDNS to retrieve reverse DNS lookup information.
Cisco Threat Grid integration	Integrate EclecticIQ Platform with Cisco Threat Grid through the Threat Grid API. You can implement the integration as an incoming feed to ingest entities, as well as an enricher to produce enrichm...
Build custom enrichers	Implement custom extensions to integrate EclecticIQ Platform with external intel providers through incoming feeds and enrichers, as well as to publish platform intel downstream in your prevention/d...
Flashpoint integration	Integrate EclecticIQ Platform with Flashpoint AggregINT, Flashpoint Blueprint, and Flashpoint Thresher through the Flashpoint API.
Intel 471 integration	Integrate EclecticIQ Platform with Intel 471 through the Intel 471 API. You can implement the integration as an incoming feed to ingest entities, as well as an enricher to produce enrichment observ...
PassiveTotal integration	Integrate EclecticIQ Platform with RiskIQ PassiveTotal to retrieve active/passive DNS, IP, domain, and malware information.
Splunk integration	EclecticIQ Platform App for Splunk Enterprise enables Splunk users to ingest large quantities of threat intelligence by integrating EclecticIQ Platform feeds with Splunk Enterprise.
VirusTotal integration	Integrate EclecticIQ Platform with VirusTotal to retrieve malware information about DNSs, IPs, domains, and files.

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

Cisco OpenDNS integration

Integrate EclecticIQ Platform with Cisco OpenDNS through OpenResolve by OpenDNS to retrieve reverse DNS lookup information.

Cisco OpenDNS OpenResolve	**integration**
Integration	OpenDNS OpenResolve
Type	enricher/API
API endpoint	`http://api.openresolve.com/{}/{}`
Input	ipv4, ipv6, domain, host
Output	Enriches the 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.

EclecticIQ Platform integrates with Cisco OpenDNS solutions to use them as intel sources. The platform integrates with the Cisco OpenDNS OpenResolve API through an enricher.

To enable the Cisco OpenDNS integration you configure the Cisco OpenDNS/OpenResolve enricher as needed, and then activate it or run it manually to poll the intel source.

Configure the enricher

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 the enrichment information. Data sources can be incoming feeds, as well as other enrichers.
- Within the selected platform data sources, the entity type(s) to augment with the enrichment information.
- The enrichers to use to fetch the 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.

Observables

Observables augment the entities they are related to by providing additional context that can help discover indirect relationships or spawn new relationships between entities.
Observables are atomic and factual: an observable represents one discrete piece of information that describes a fact. For example, an IP address, a hash value, the name of a location or an actor.

Configure enricher tasks

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

- On the top navigation bar click ✚ **> Data management > Dataset > Enrichment** .

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Data management**.
- On the left-hand navigation sidebar click **Enrichment**.
- Click the enricher you want to configure or modify.
- On the enricher detail page, click the **Edit** button.

> ✔ On the forms, input fields marked with an asterisk are required.

The Cisco OpenDNS 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.

Configure enricher rules

Add enricher rules

To add a new enricher rule, do the following:

- On the top navigation bar click ✚ **> Rules > Enrichment**.

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.

- On the left-hand navigation sidebar click **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 **+ Rule** button.

> ✔ On the forms, input fields marked with an asterisk are required.

On the **Rules > Enrichment > Create** 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 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 creation work.

Edit enricher rules

To edit enricher rules, do the following:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.

- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 detail pane, click **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**.

> ✔ On the forms, 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 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.
- Click **Delete** on the rule detail pane.

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.

Run the enricher

Automatically

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

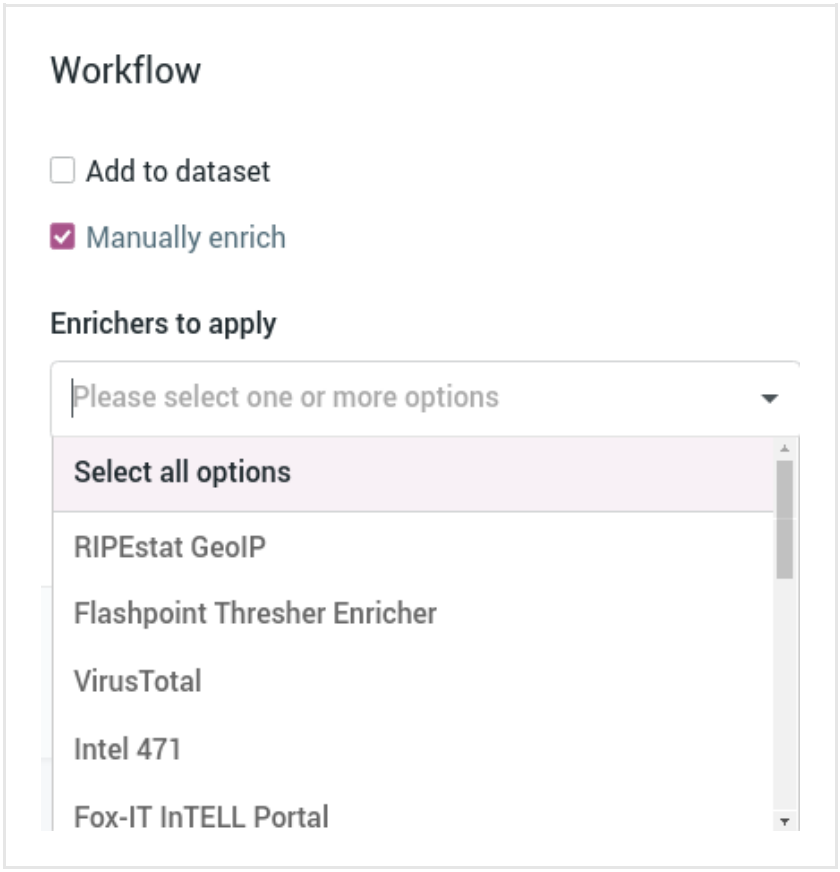
Rules give you control over the type of information you want to retrieve or exclude, and what you want to do with it. You can assign one or more enricher sources to specific observable types. You can set multiple filters to cover usage scenarios as needed. You can then examine the returned enrichment 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**.

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.

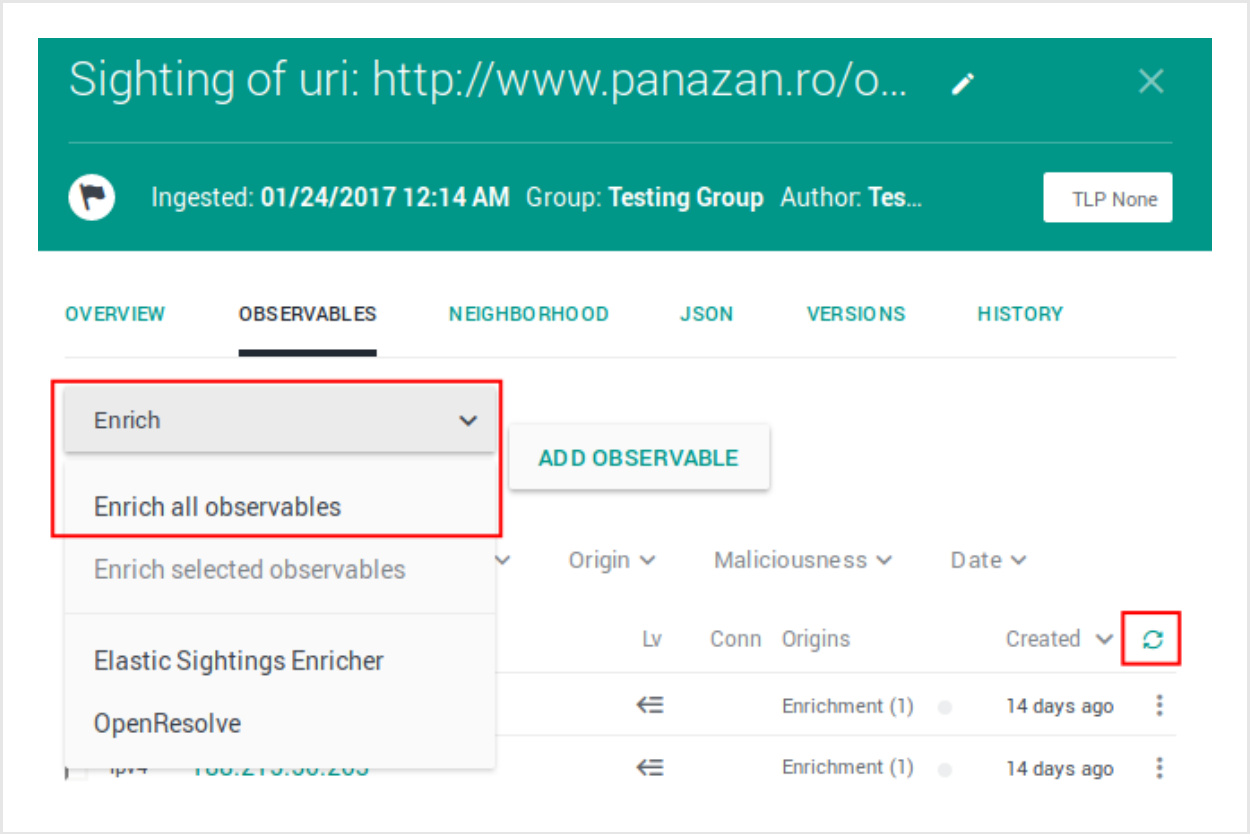
- On the entity detail pane, click **Observables**.
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.

To manually enrich the entity observables:

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

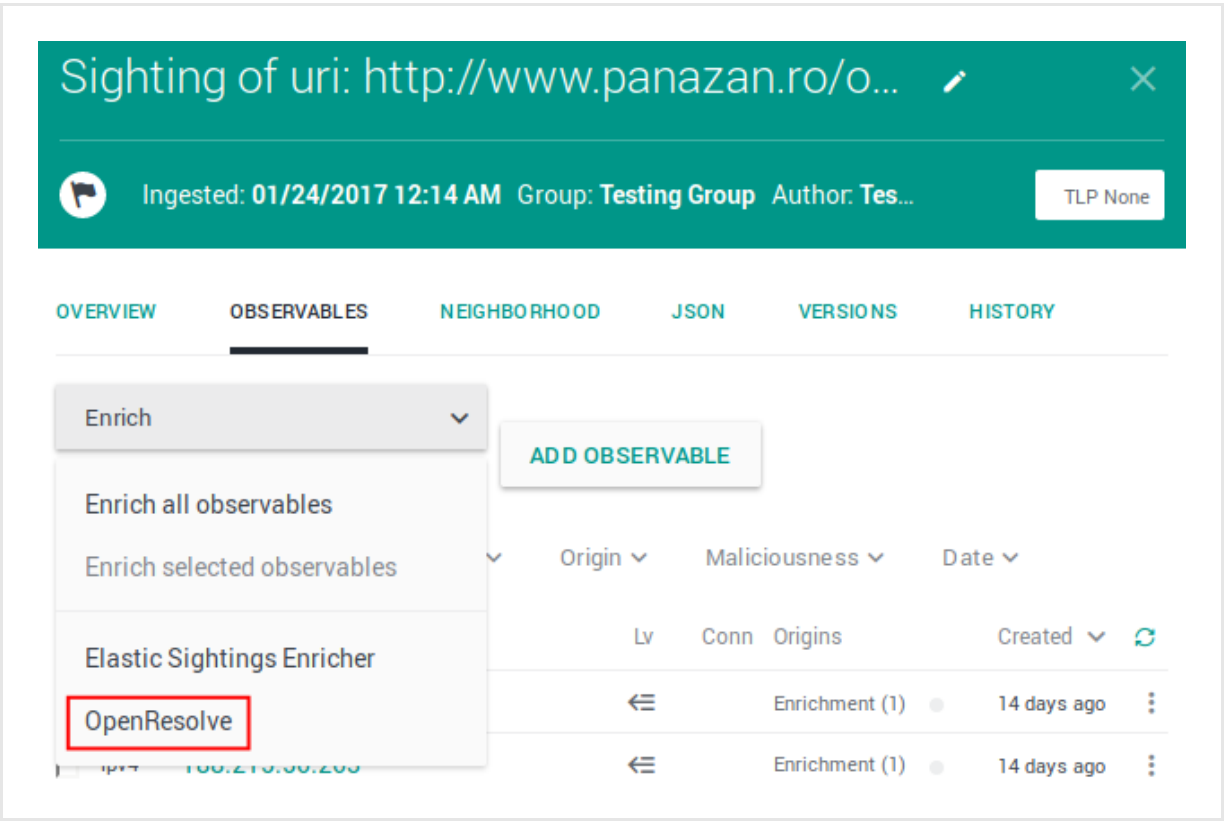
Alternatively:

- From the **Enrich** drop-down menu, select **Enrich all observables**.
- 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:

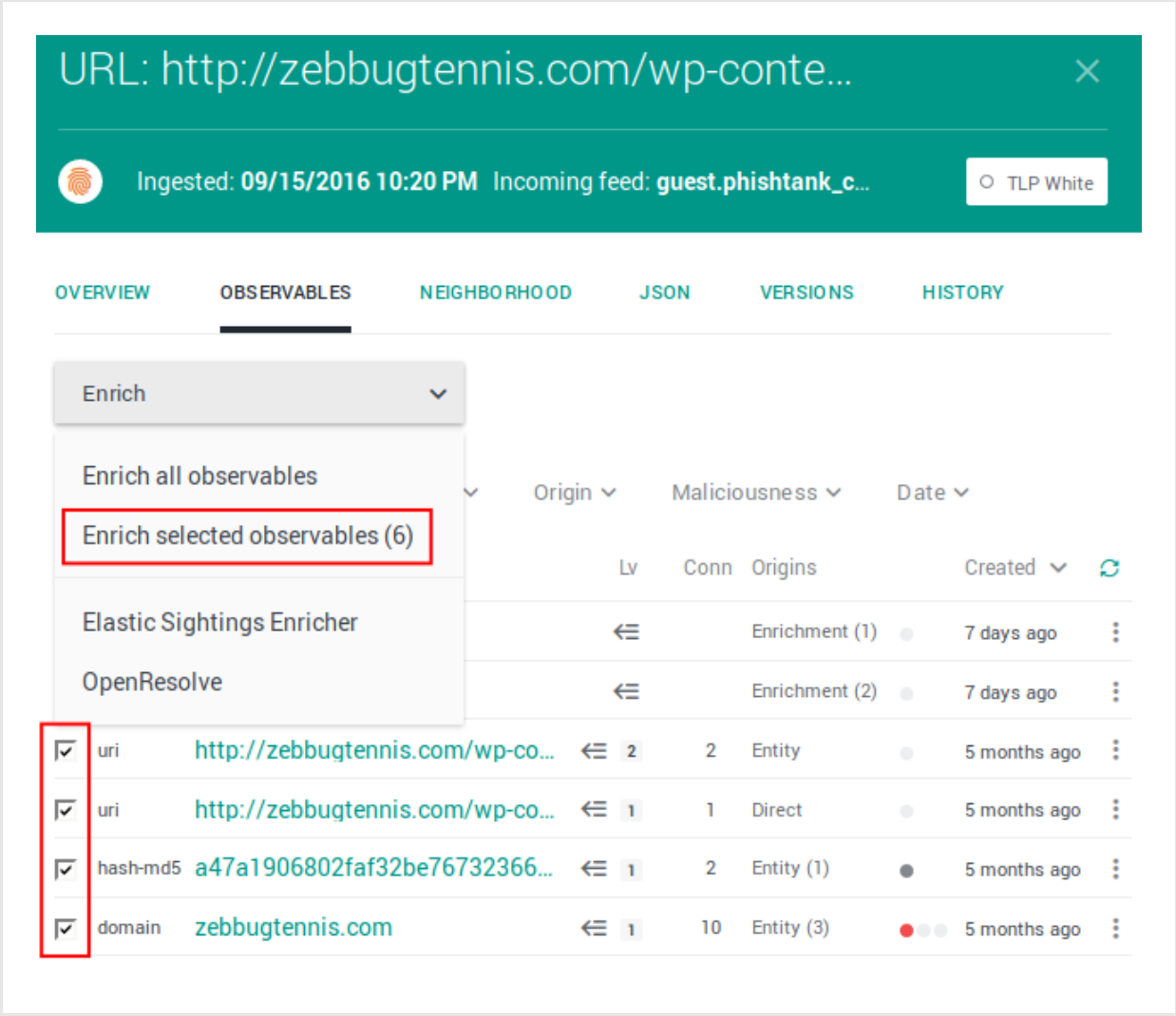
- Select it from the **Enrich** drop-down menu, and then click it.
- The platform polls the specified enricher for the entity, and it enriches all the entity observables with the retrieved data.



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 selected observables**.
- The platform polls all applicable enrichers for the entity, and it enriches the selected entity observables with the retrieved data.



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

The Cisco OpenDNS enricher can take the following observable types as input:

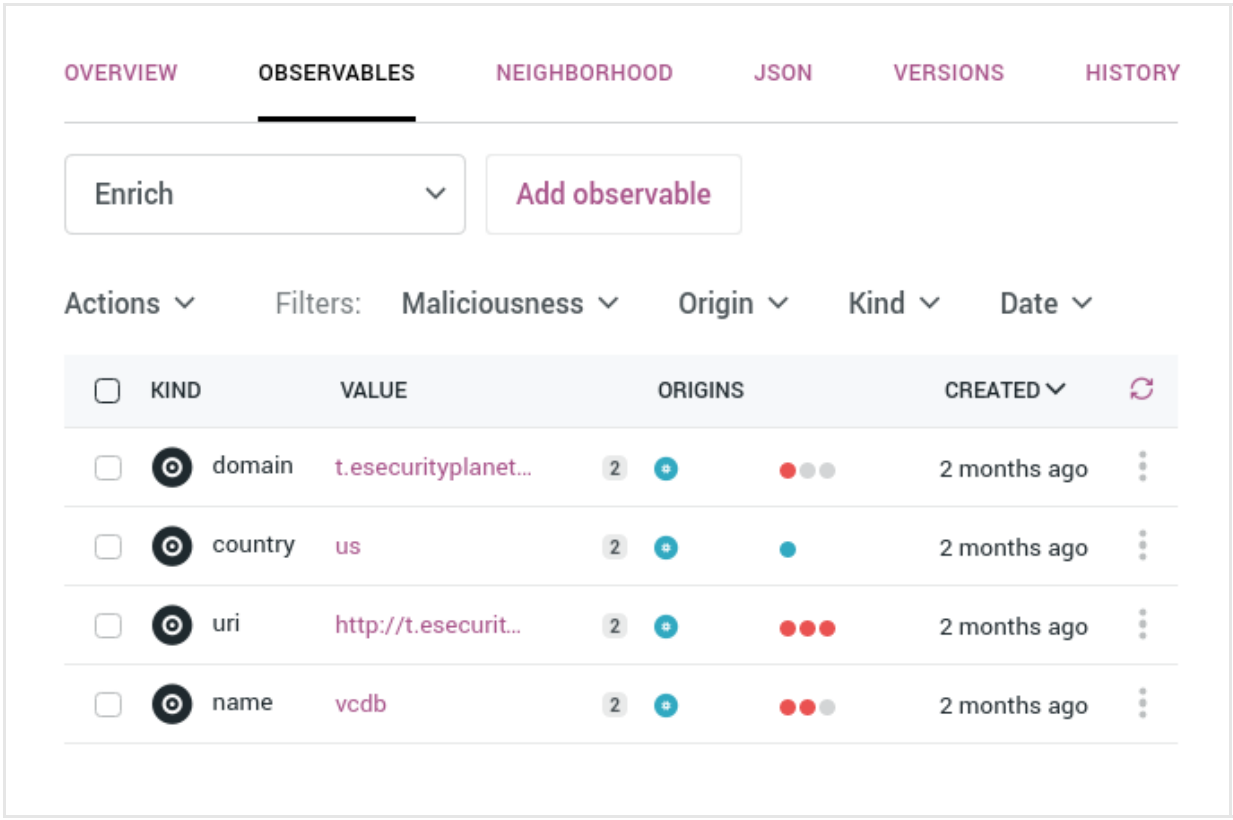
- *ipv4, ipv6, domain, host*

The enricher uses these input data types to look for additional information to enrich existing observables with. Any entity types supporting these observable types can be enriched with OpenDNS OpenResolve.

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**.

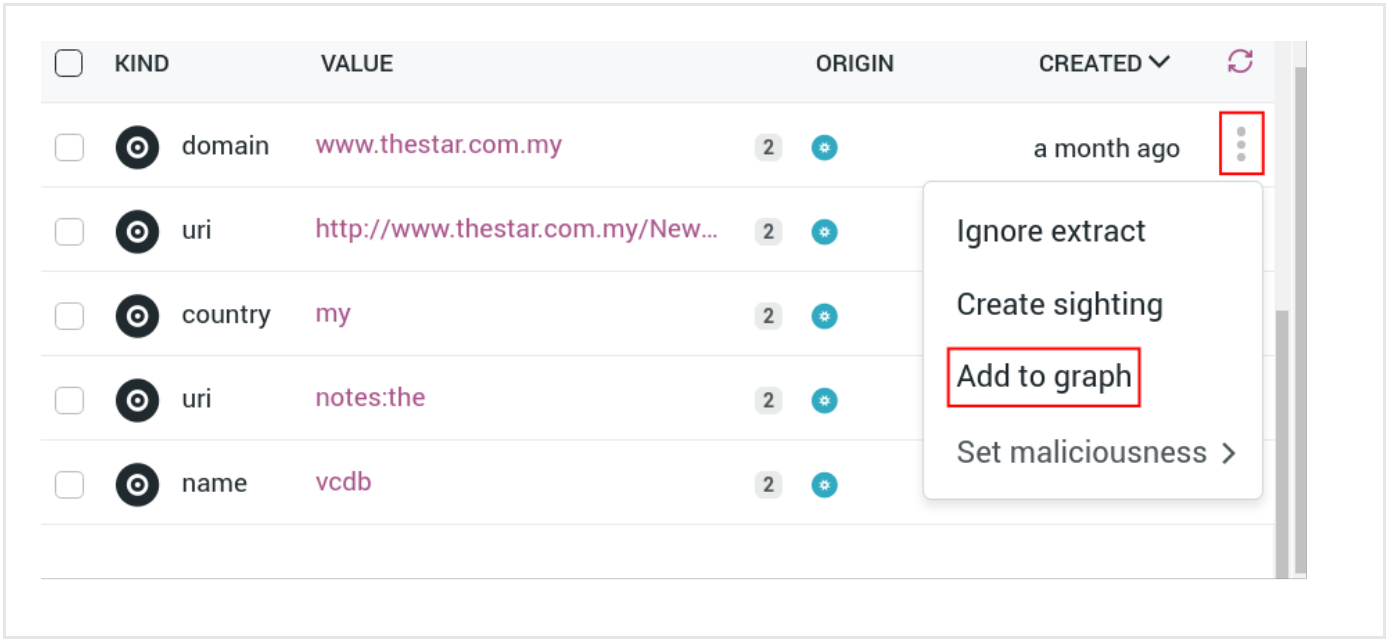
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.



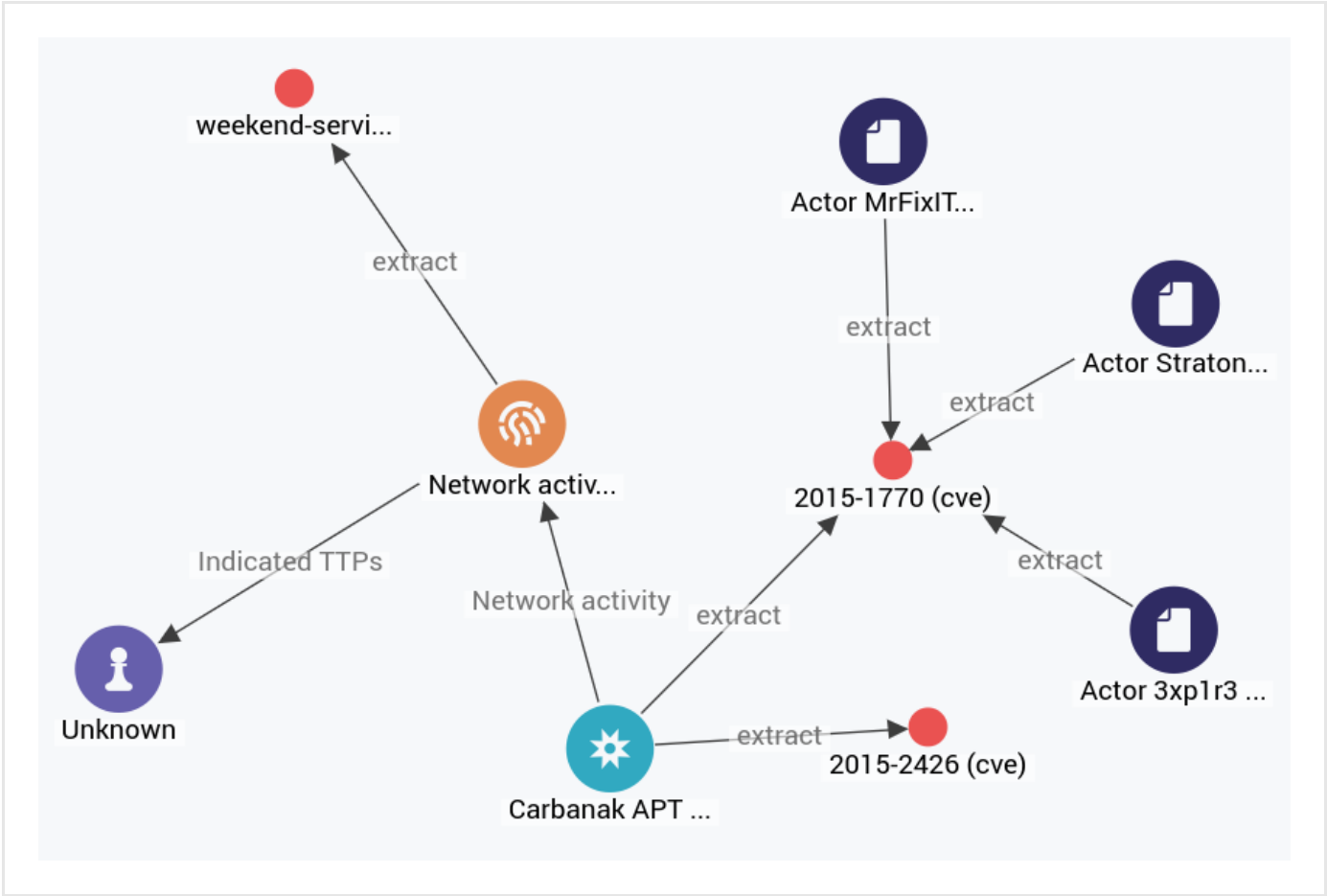
Review enrichment observables on the graph

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

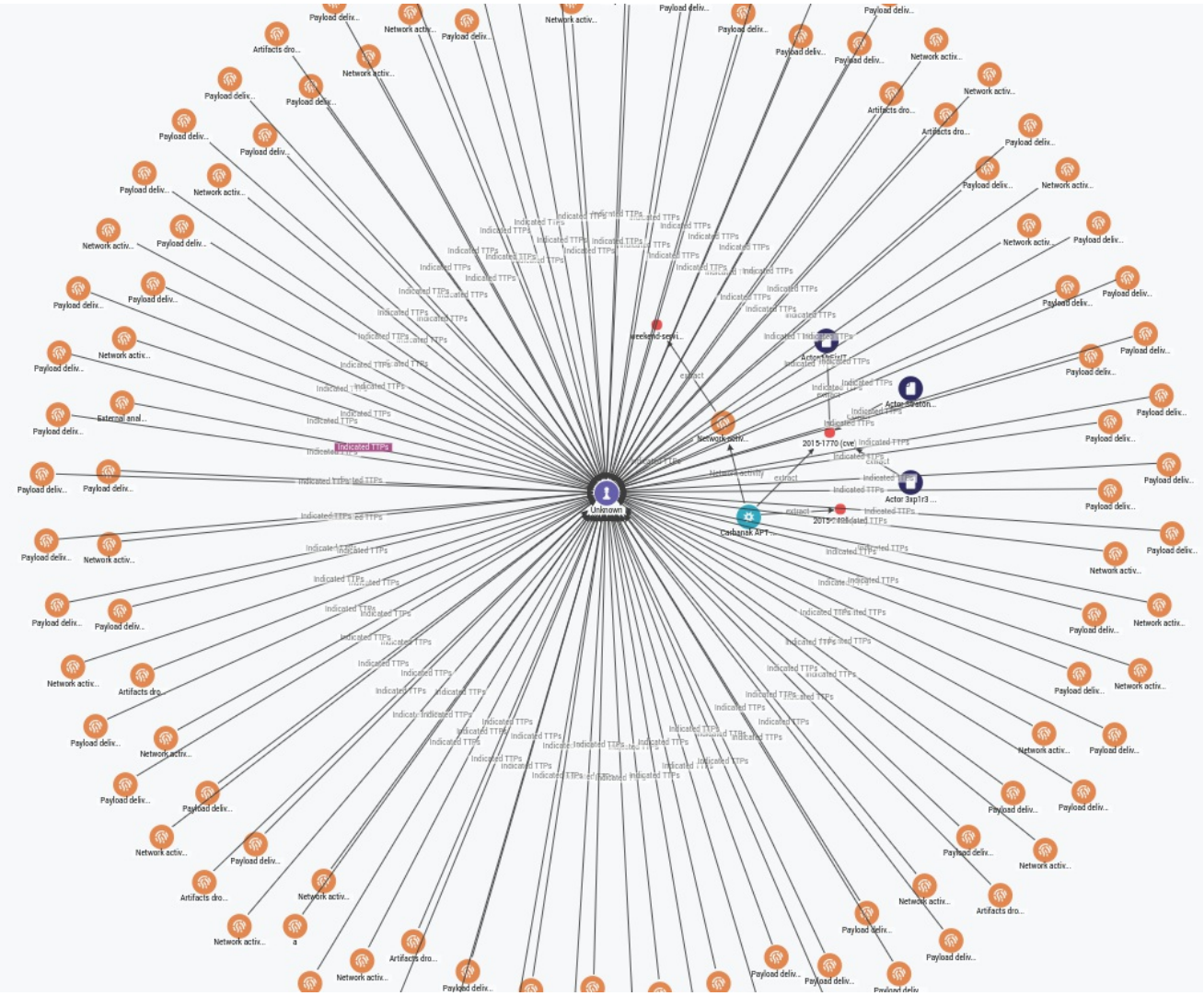
- On the row corresponding to the observable you want to load onto the graph, click the ⋮ icon, and then select **Add to graph**.



- 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** .

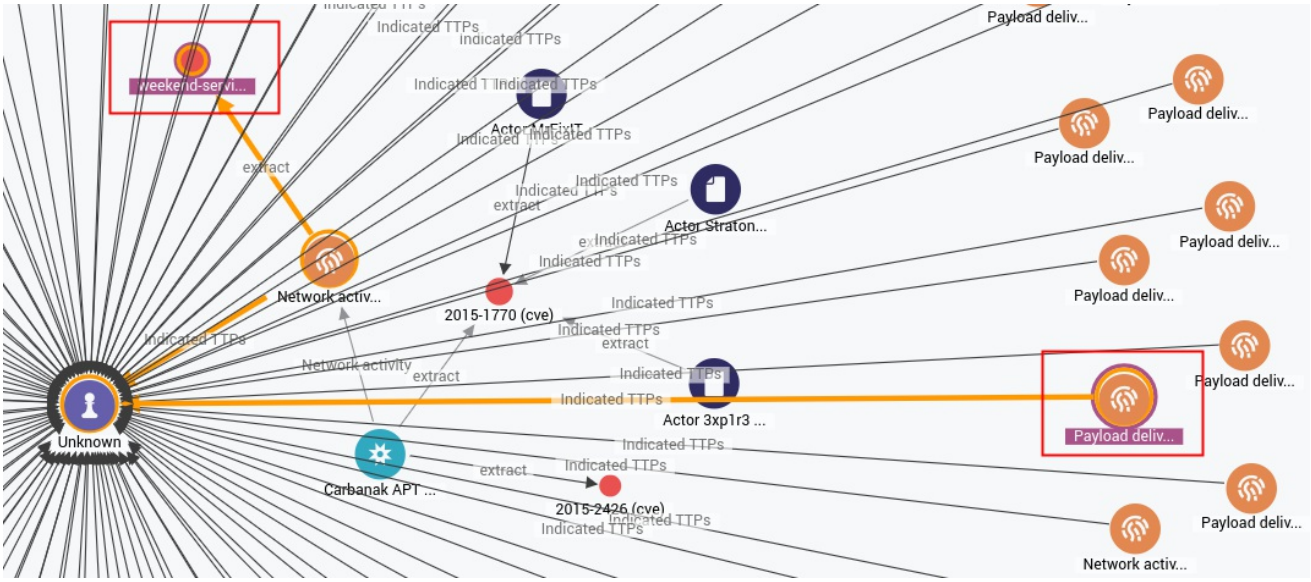


- Right-click an extract or an entity for further inspection and from the context menu select **Load entities > All**, **Load observables > All** or **Load entities by extract > All** .



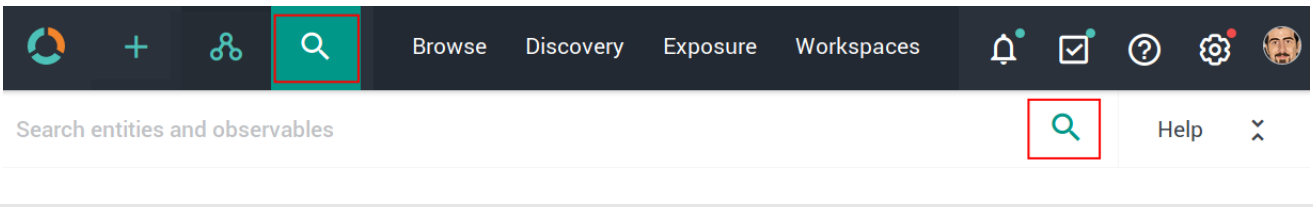
To see how entities, observables and enrichment observables are connected, and to inspect relationships between distant items, do the following:

- **CTRL** + click two nodes on the graph to select them.
- Right-click either selected node, and from the context menu select **Find path** to query the graph database about the existence of a path between the nodes, or **Show path** to highlight asn existing path on the graph.
- If a path does exist, the selected nodes and all the intermediate ones are highlighted on the graph to show the path that links them.



Search for enrichment observables

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

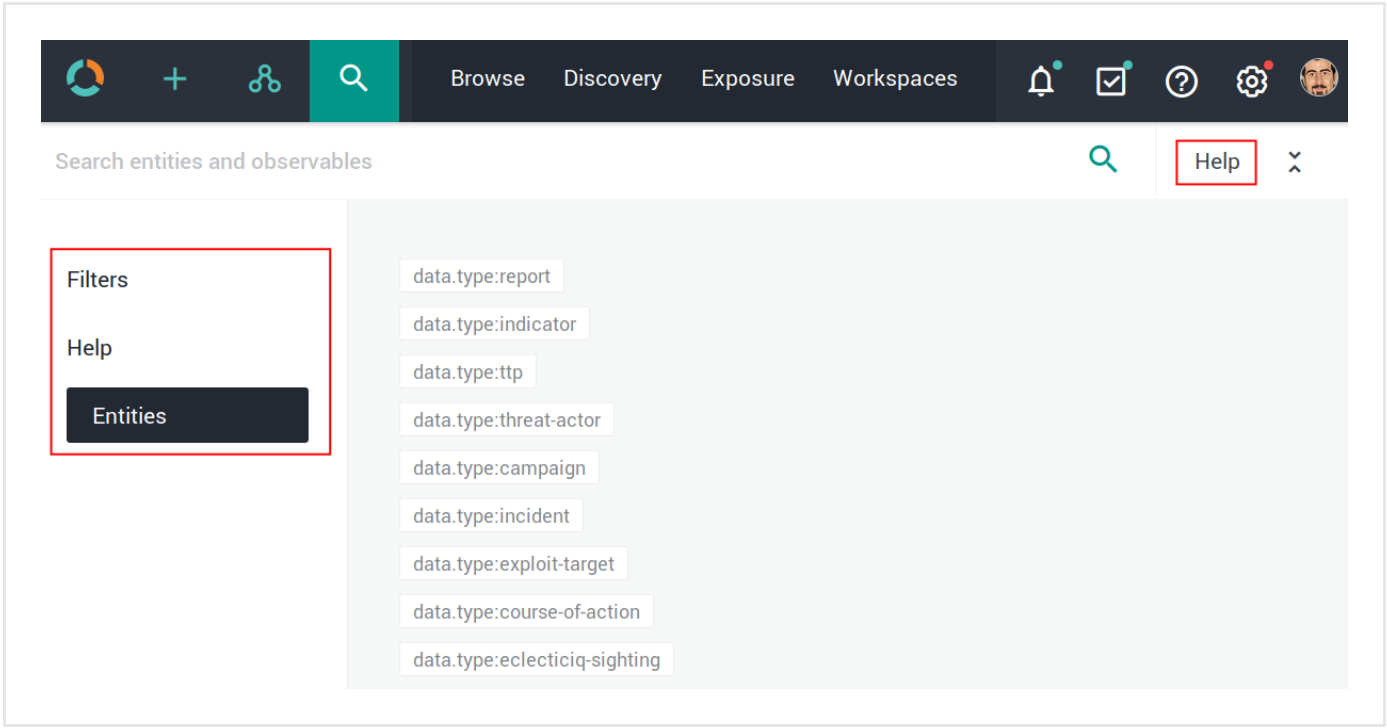


Enter search terms and search queries, and then press **ENTER** or click the search icon to run the search. Searches you run through this search box are executed platform-wide.

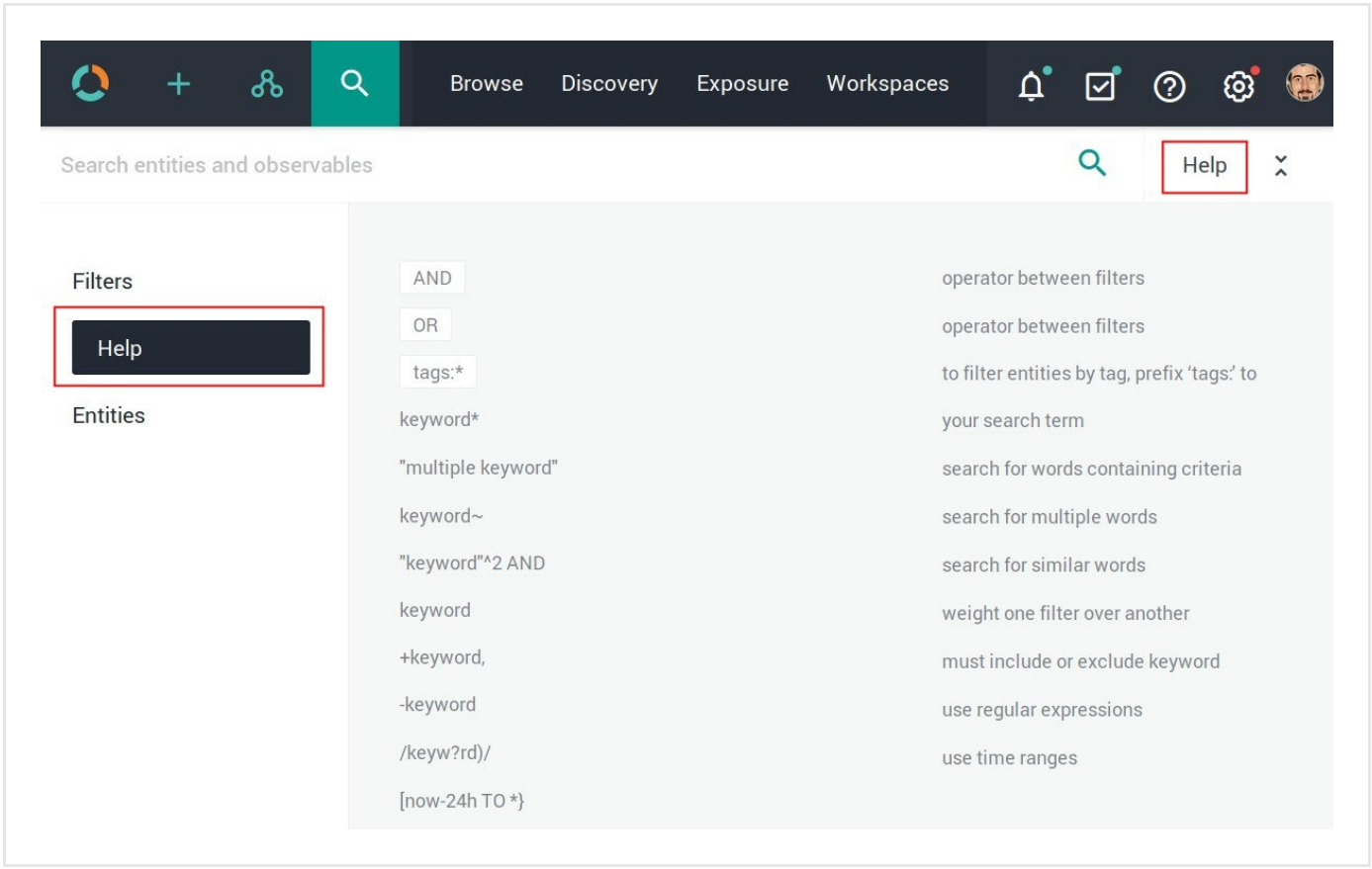
> The search functionality uses **Elasticsearch query syntax** (https://www.elastic.co/guide/en/elasticsearch/reference/current/full-text-queries.html).

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.



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



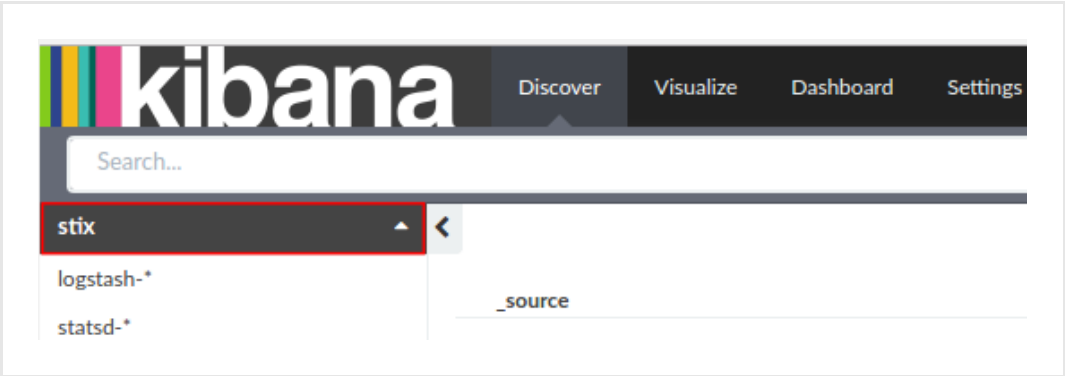
Use operators to combine multiple quick filters and create a more complex search query.
Example:

enrichment_extracts.kind:domain AND enrichment_extracts.meta.classification:high

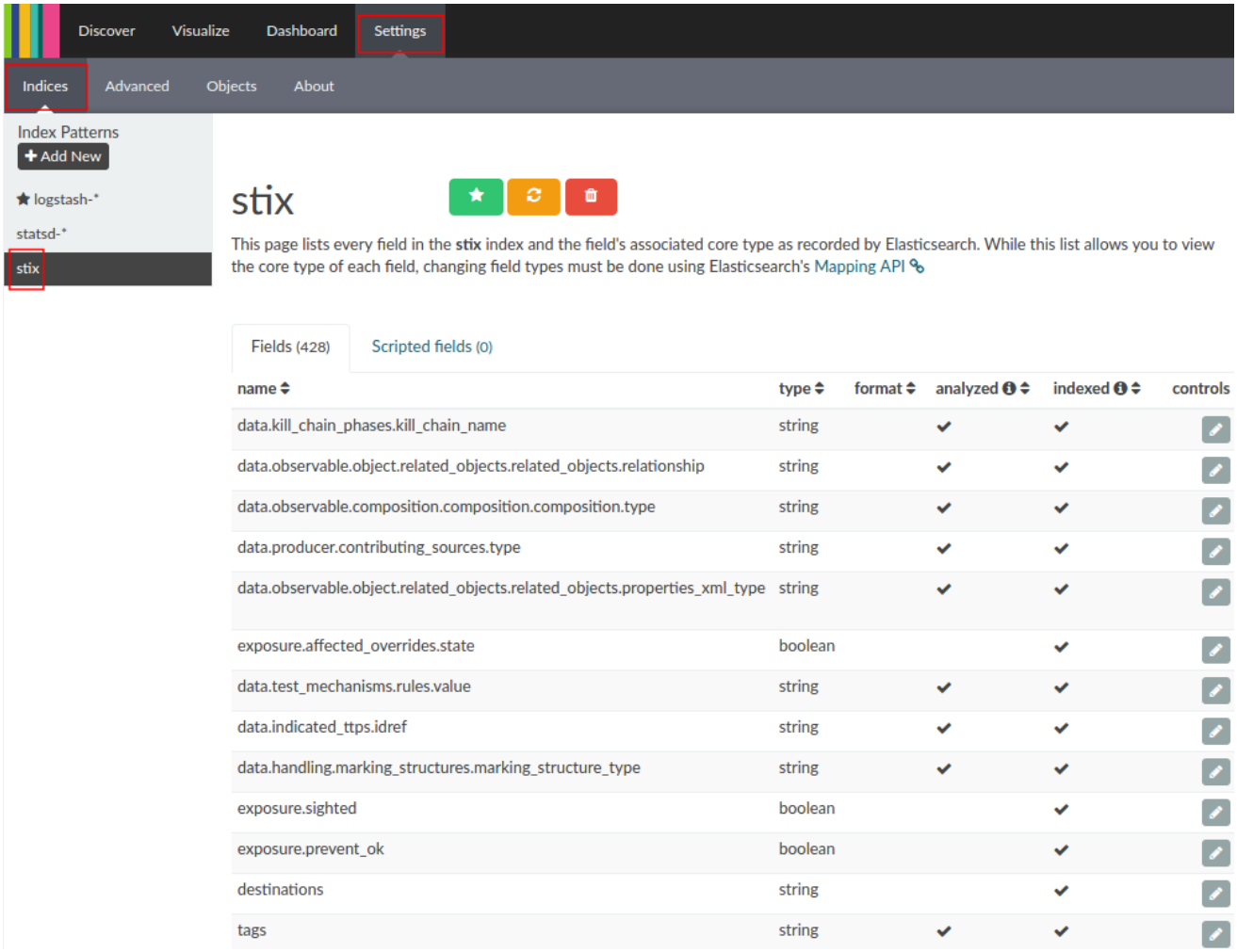
Field	Description	Example
enrichment_extracts.id	string — The alphanumeric ID string that uniquely identifies the enrichment observable.	`01h12x45-01q2-1234-od01-123456h78h90`
enrichment_extracts.kind	string — The enrichment observable data type.	`domain`
enrichment_extracts.meta.blacklisted	Boolean — An observable is blacklisted when it is included in the results returned by an *ignore* extraction rule. Allowed values: `true`, `false`.	`true`
enrichment_extracts.meta.classification	string — This value is defined in **Rules** by selecting appropriate options under **Action** and **Confidence**. Allowed classification metadata values are `good`, `bad`, and `unknown`.	`good`
enrichment_extracts.meta.confidence	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 `low`, `medium`, and `high`.	`high`
enrichment_extracts.value	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:

- Sign in to the platform with your user credentials.
- To access Kibana, in the web browser address bar enter a URL with the following format: `<platform_host>/api/kibana/app/kibana#/`.
  Keep the trailing `/`.
  Example: `https://platform.host.com/api/kibana/app/kibana#/`
- Select the **stix** index field:



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



Cisco Threat Grid integration

Integrate EclecticIQ Platform with Cisco Threat Grid through the Threat Grid API. You can implement the integration as an incoming feed to ingest entities, as well as an enricher to produce enrichment observables that augment entity intel value.

Cisco Threat Grid	**integration**
Integration	Cisco Threat Grid
Type	enricher/API, incoming feed/API
API endpoint	`https://panacea.threatgrid.com/api/v2/`
Input	ipv4, ipv6, domain, host, uri, hash-md5, hash-sha1, hash-sha256, hash-sha512, winregistry
Output	Indicators with embedded observables representing IOCs (integration as incoming feed). Enriches the 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. (integration as enricher).
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.

EclecticIQ Platform integrates with Cisco Threat Grid to use it as an intel provider. The platform integrates with the Cisco Threat Grid API in the following ways:

- By configuring the Threat Grid API as an incoming feed intel provider.
  When it is configured as an incoming feed, the Cisco Threat Grid integration provides data that is ingested and stored as entities.
- By configuring the Threat Grid API as an enricher.
  When it is configured as an enricher, the Cisco Threat Grid integration provides enrichment data that produces entity observables.

Configure Cisco Threat Grid as an incoming feed

Configure the general options

✔ On the forms, input fields marked with an asterisk are required.

- On the top navigation bar, select **Configuration**, and then **Incoming feeds**.
- On the top-left corner of the page click the ✪ icon to open the incoming feed editor.

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-right corner of the screen, click the **+ Incoming feed** button.
- On the **✚ > Data management > Incoming feeds > Create** 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)`.
- **Extraction ignore levels**: from the drop-down menu select at least one option if you want to *exclude extracted content* from the ingested data.
  If you select at least one value, the filter excludes extracted data from ingestion, based on the direct or indirect relationship the data has with the entity it refers to.
  In other words, the filter ignores specific data, based on the data location in the entity data structure:
  - **Extraction ignore levels — 1**: the extracted data is inside a CybOX object. The ingestion process ignores and it does not include extracted data found inside observable CybOX objects embedded in STIX indicators.
  - **Extraction ignore levels — 2**: the extracted data is outside a CybOX object. The ingestion process ignores and it does not include extracted data found inside STIX fields. For example, STIX headers, titles or references.
- From the **Transport type** drop-down list, select **ThreatGRID API**.

Configure the transport type

Cisco Threat Grid API

Set a schedule

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

- **None**: no schedule is defined. You need to manually trigger the task to ingest or to publish data through an incoming or an outgoing feed, respectively.
- **Minute**: the feed task runs automatically 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.
- **Hour**: the feed task runs automatically every hour.
  You define how long in minutes after the beginning of an hour the task should run from the corresponding drop-down menu.
- **Day**: the feed task runs automatically once a day.
  You define the time of the day when the task should run from the corresponding drop-down menu.
- **Week**: the feed task runs automatically once a week.
  You define the day of the week and time of the day when the task should run from the corresponding drop-down menu.

- **Month**: the feed task runs automatically once a month.
  You define the day of the calendar month and time of the day when the task should run from the corresponding drop-down menu.
  Keep in mind that not all months of the year have 31 days.

Set a TLP override

Override TLP overwrites the **TLP** `(https://www.us-cert.gov/tlp)` color code associated to 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 properties:

- **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 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 creation work.

Configure Cisco Threat Grid as an enricher

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 the enrichment information. Data sources can be incoming feeds, as well as other enrichers.
- Within the selected platform data sources, the entity type(s) to augment with the enrichment information.
- The enrichers to use to fetch the 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.

Observables

Observables augment the entities they are related to by providing additional context that can help discover indirect relationships or spawn new relationships between entities.
Observables are atomic and factual: an observable represents one discrete piece of information that describes a fact. For example, an IP address, a hash value, the name of a location or an actor.

Configure enricher tasks

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

- On the top navigation bar click ✚ **> Data management > Dataset > Enrichment** .

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Data management**.
- On the left-hand navigation sidebar click **Enrichment**.
- Click the enricher you want to configure or modify.

- On the enricher detail page, click the **Edit** button.

> ✔ On the forms, input fields marked with an asterisk are required.

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 intelligence 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 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 with **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 *high confidence* value.
  After completing the sample analysis, enriched observables with a *higher* threat score than the specified value are flagged with **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 with **Malicious - Medium confidence**.
- Click **Save** to store your changes, or **Cancel** to discard them.

Configure enricher rules

Add enricher rules

To add a new enricher rule, do the following:

- On the top navigation bar click **✚ > Rules > Enrichment**.

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 **+ Rule** button.

> ✔ On the forms, input fields marked with an asterisk are required.

On the **Rules > Enrichment > Create** 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 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 creation work.

Edit enricher rules

To edit enricher rules, do the following:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 detail pane, click **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**.

> ✔ On the forms, 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 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.

- Click **Delete** on the rule detail pane.

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.

Run the enricher

Automatically

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

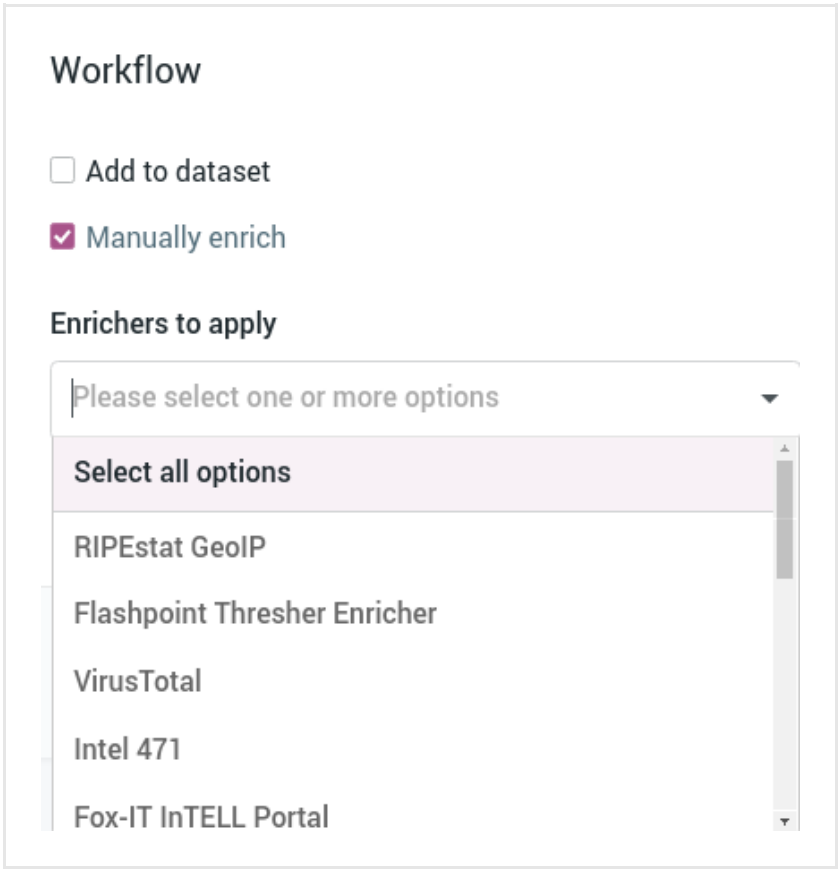
Rules give you control over the type of information you want to retrieve or exclude, and what you want to do with it. You can assign one or more enricher sources to specific observable types. You can set multiple filters to cover usage scenarios as needed. You can then examine the returned enrichment 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**.

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.

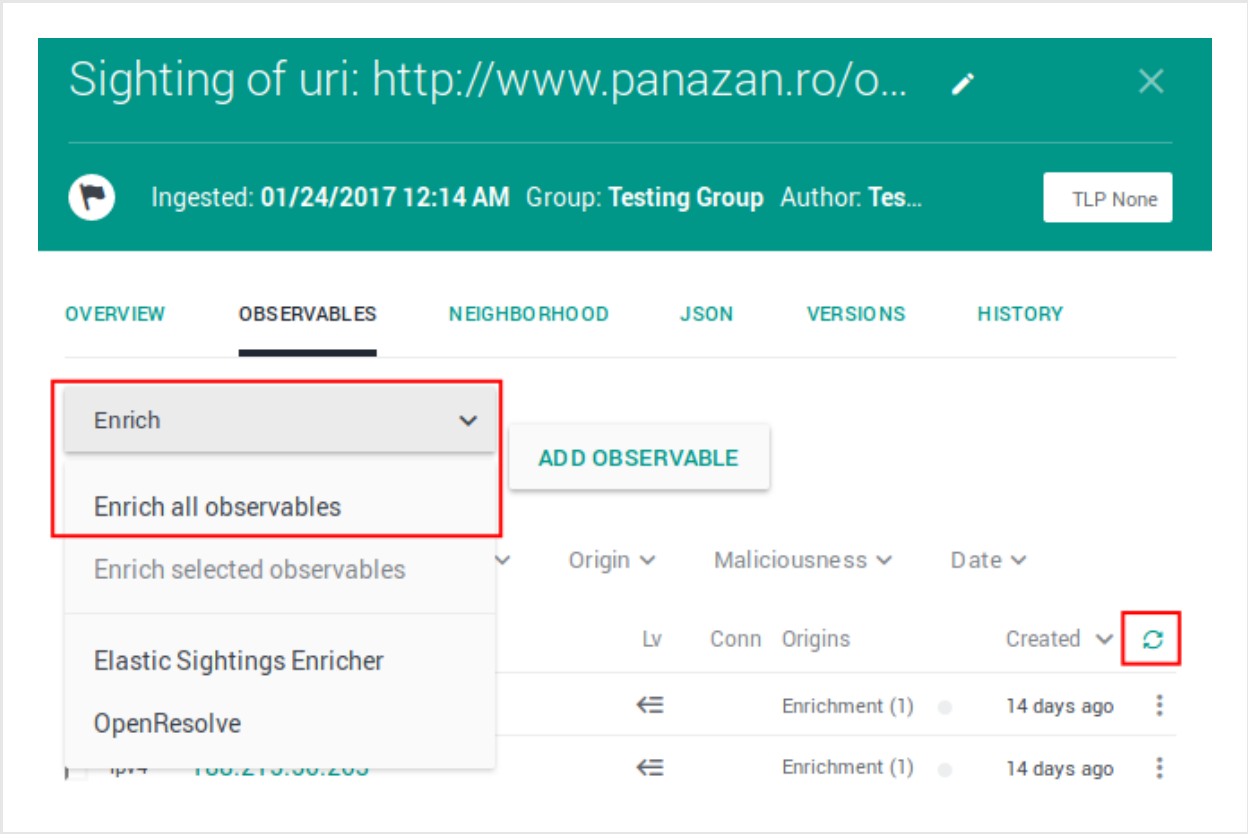
- On the entity detail pane, click **Observables**.
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.

To manually enrich the entity observables:

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

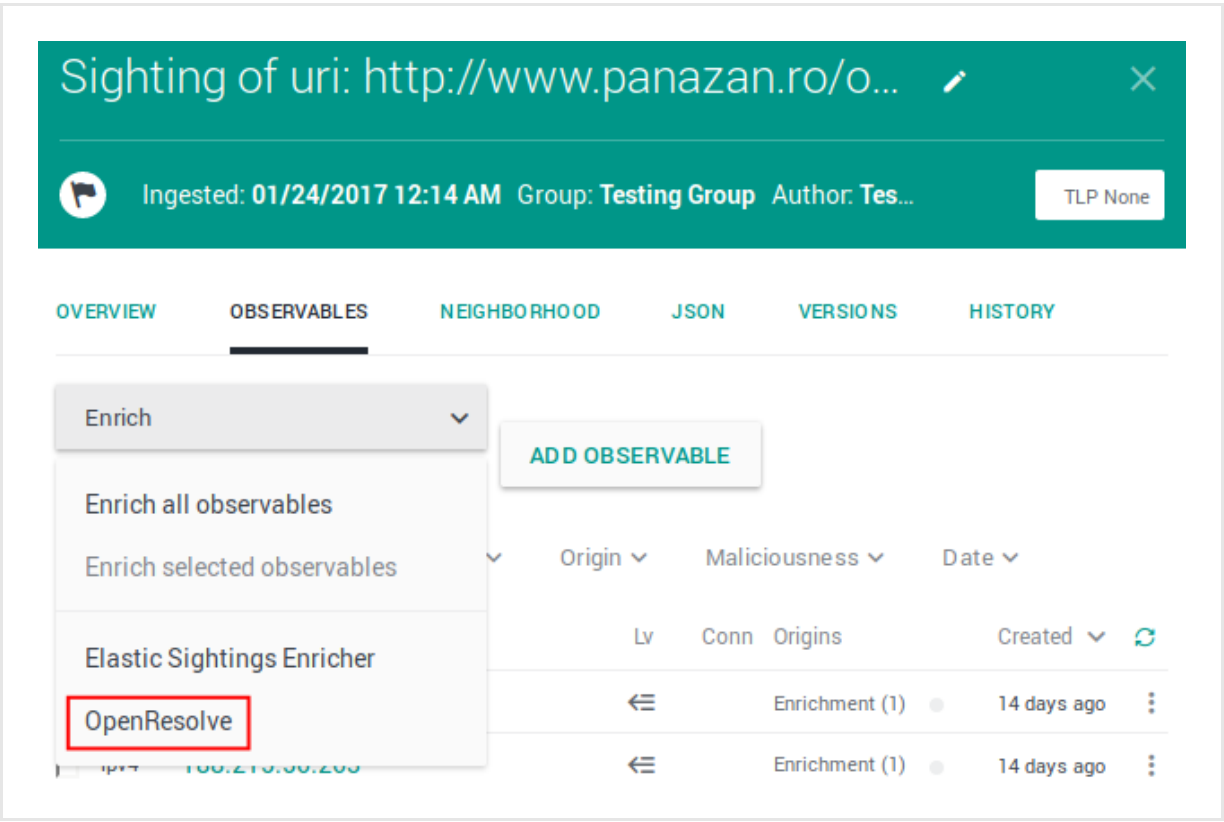
Alternatively:

- From the **Enrich** drop-down menu, select **Enrich all observables**.
- 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:

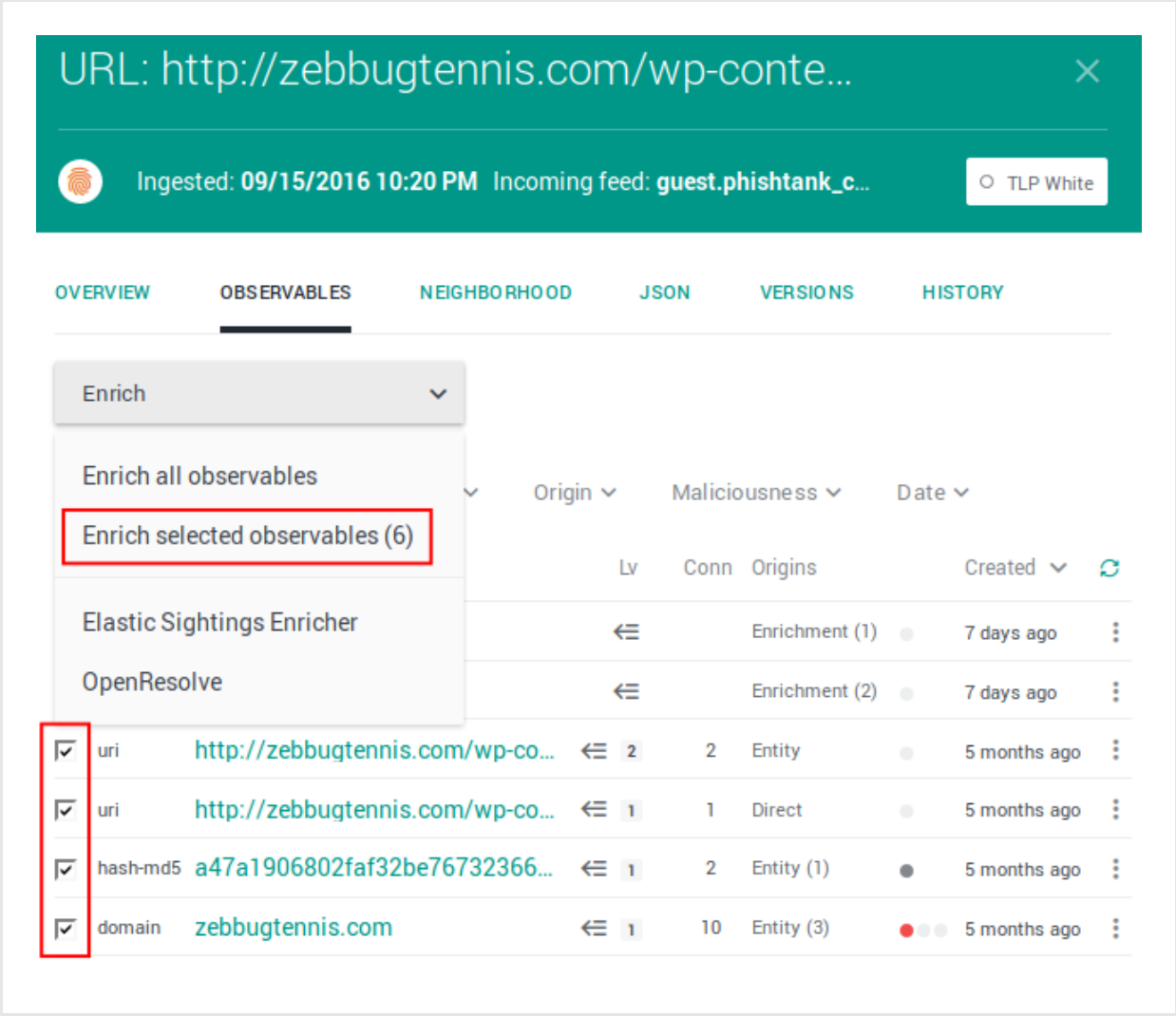
- Select it from the **Enrich** drop-down menu, and then click it.
- The platform polls the specified enricher for the entity, and it enriches all the entity observables with the retrieved data.



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 selected observables**.
- The platform polls all applicable enrichers for the entity, and it enriches the selected entity observables with the retrieved data.



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

The Cisco Threat Grid enricher can take the following observable types as input:

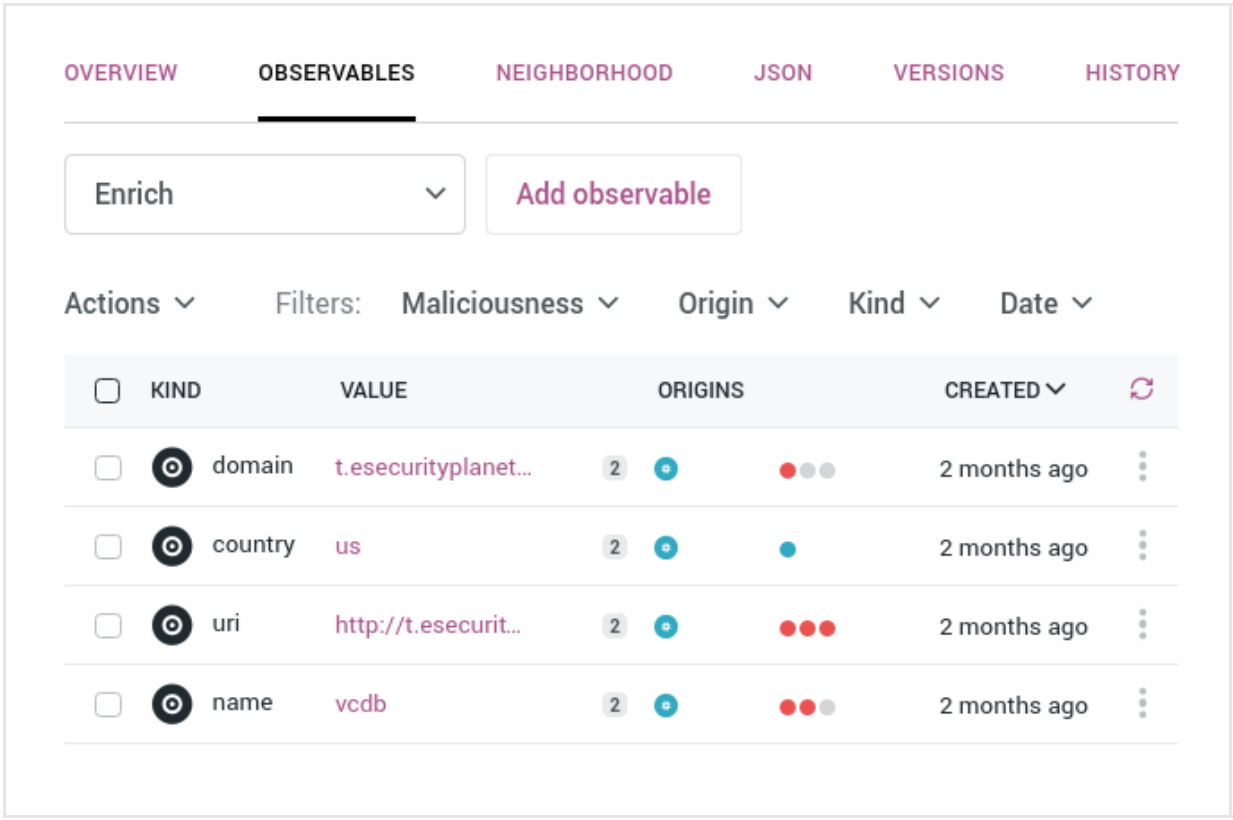
- *ipv4, ipv6, domain, host, uri, hash-md5, hash-sha1, hash-sha256, hash-sha512, winregistry*

The enricher uses these input data types to look for additional information to enrich existing observables with. Any entity types supporting these observable types can be enriched with Cisco Threat Grid.

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**.

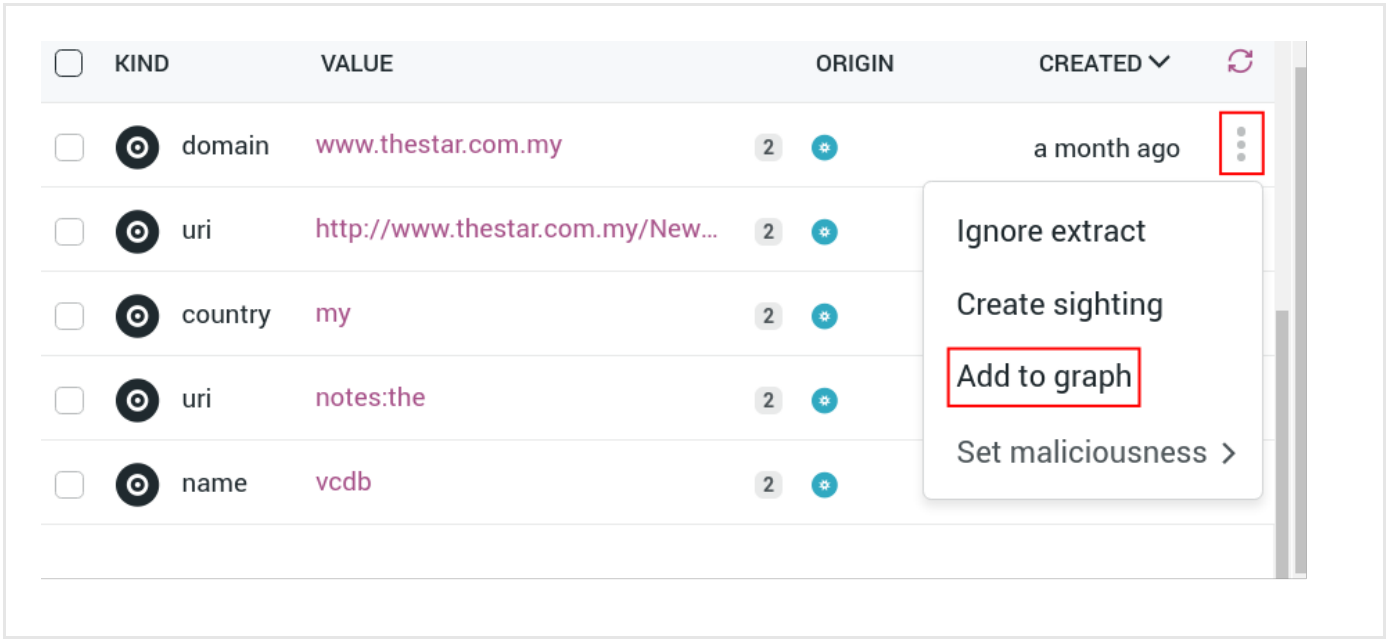
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.



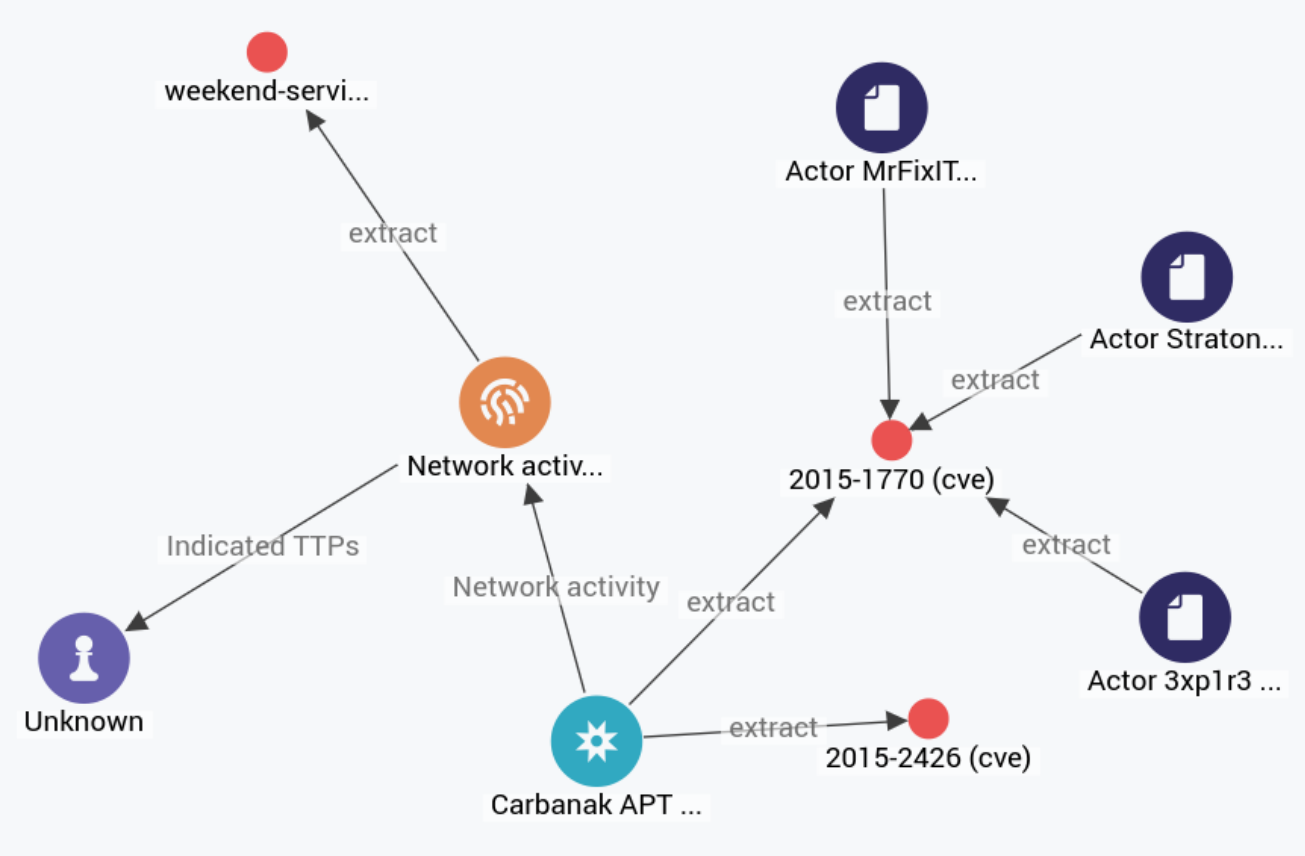
Review enrichment observables on the graph

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

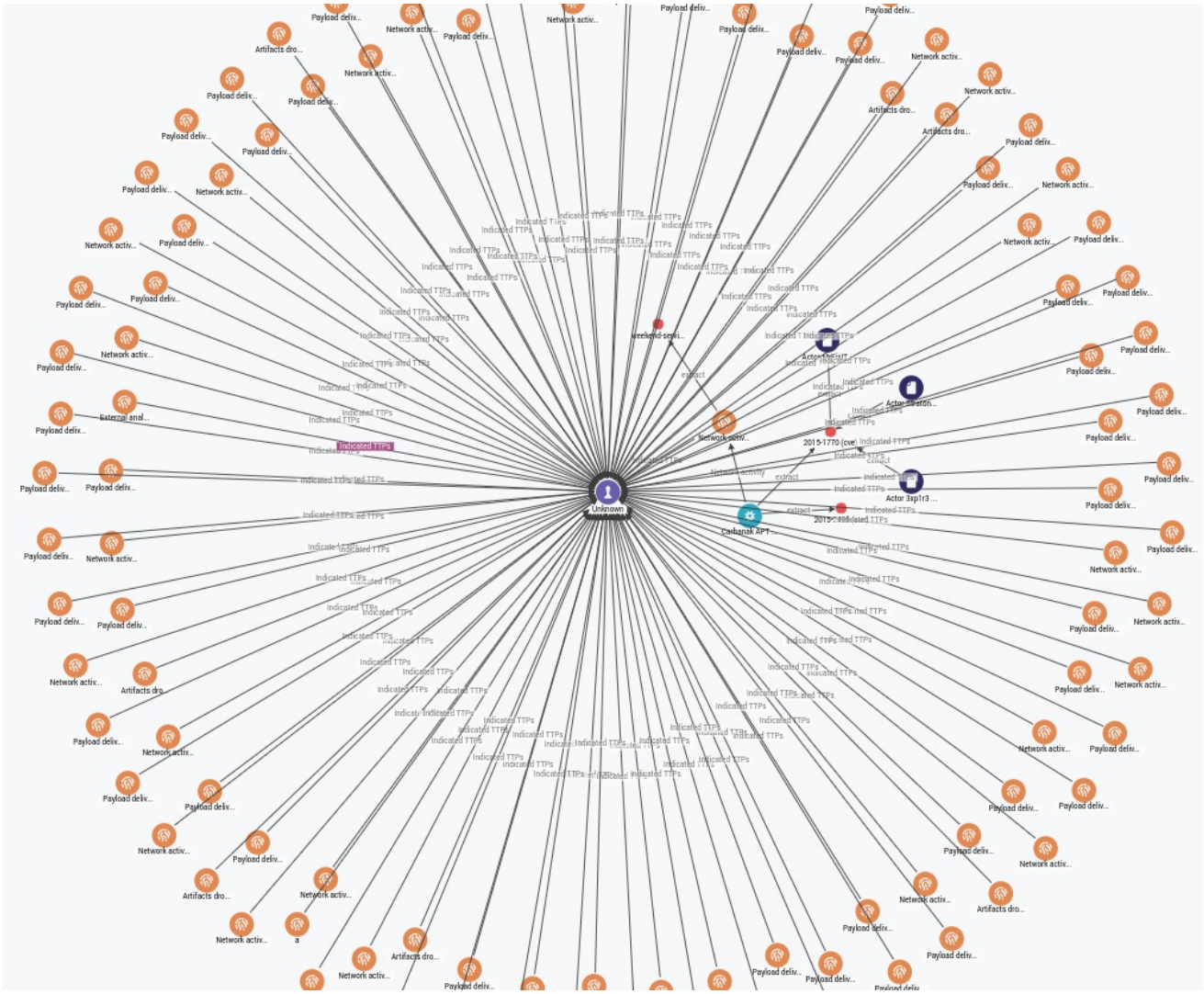
- On the row corresponding to the observable you want to load onto the graph, click the ⋮ icon, and then select **Add to graph**.



- 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** .

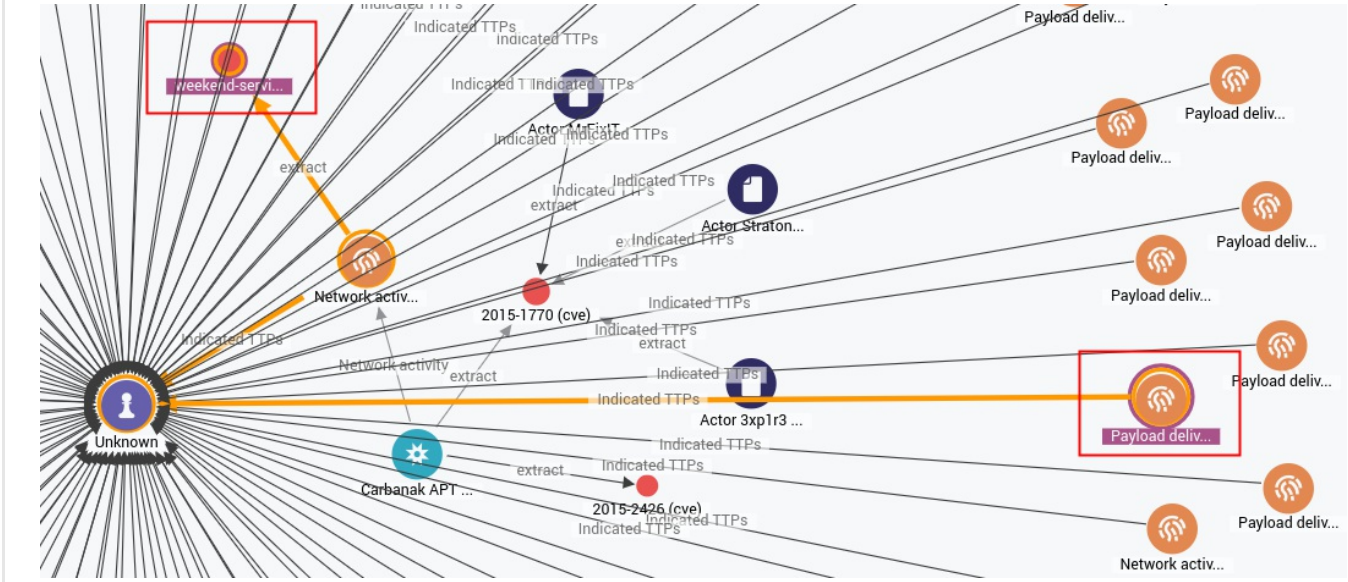


- Right-click an extract or an entity for further inspection and from the context menu select **Load entities > All**, **Load observables > All** or **Load entities by extract > All** .



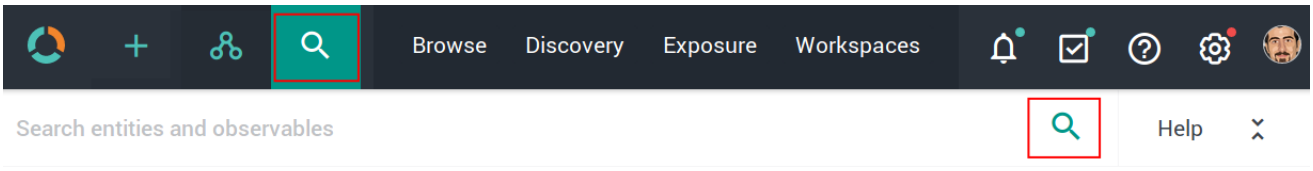
To see how entities, observables and enrichment observables are connected, and to inspect relationships between distant items, do the following:

- **CTRL** + click two nodes on the graph to select them.
- Right-click either selected node, and from the context menu select **Find path** to query the graph database about the existence of a path between the nodes, or **Show path** to highlight asn existing path on the graph.
- If a path does exist, the selected nodes and all the intermediate ones are highlighted on the graph to show the path that links them.



Search for enrichment observables

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

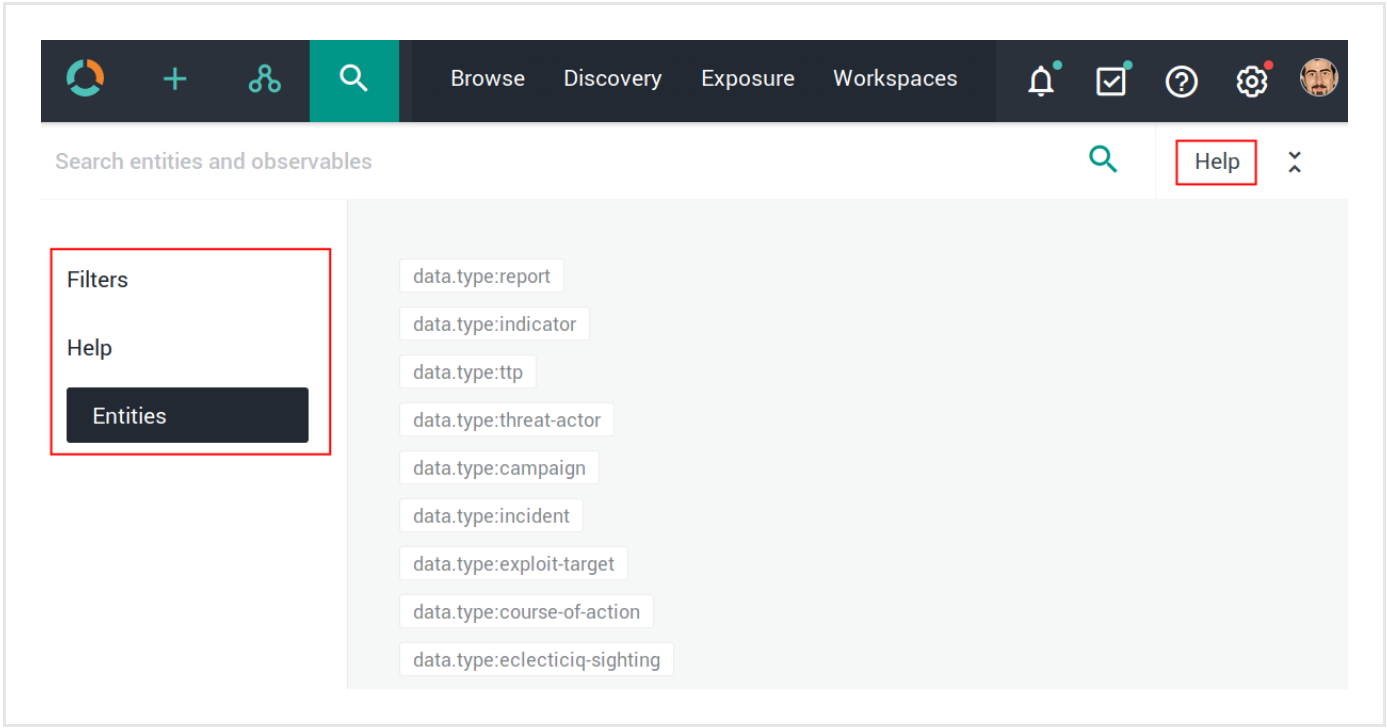


Enter search terms and search queries, and then press **ENTER** or click the search icon to run the search. Searches you run through this search box are executed platform-wide.

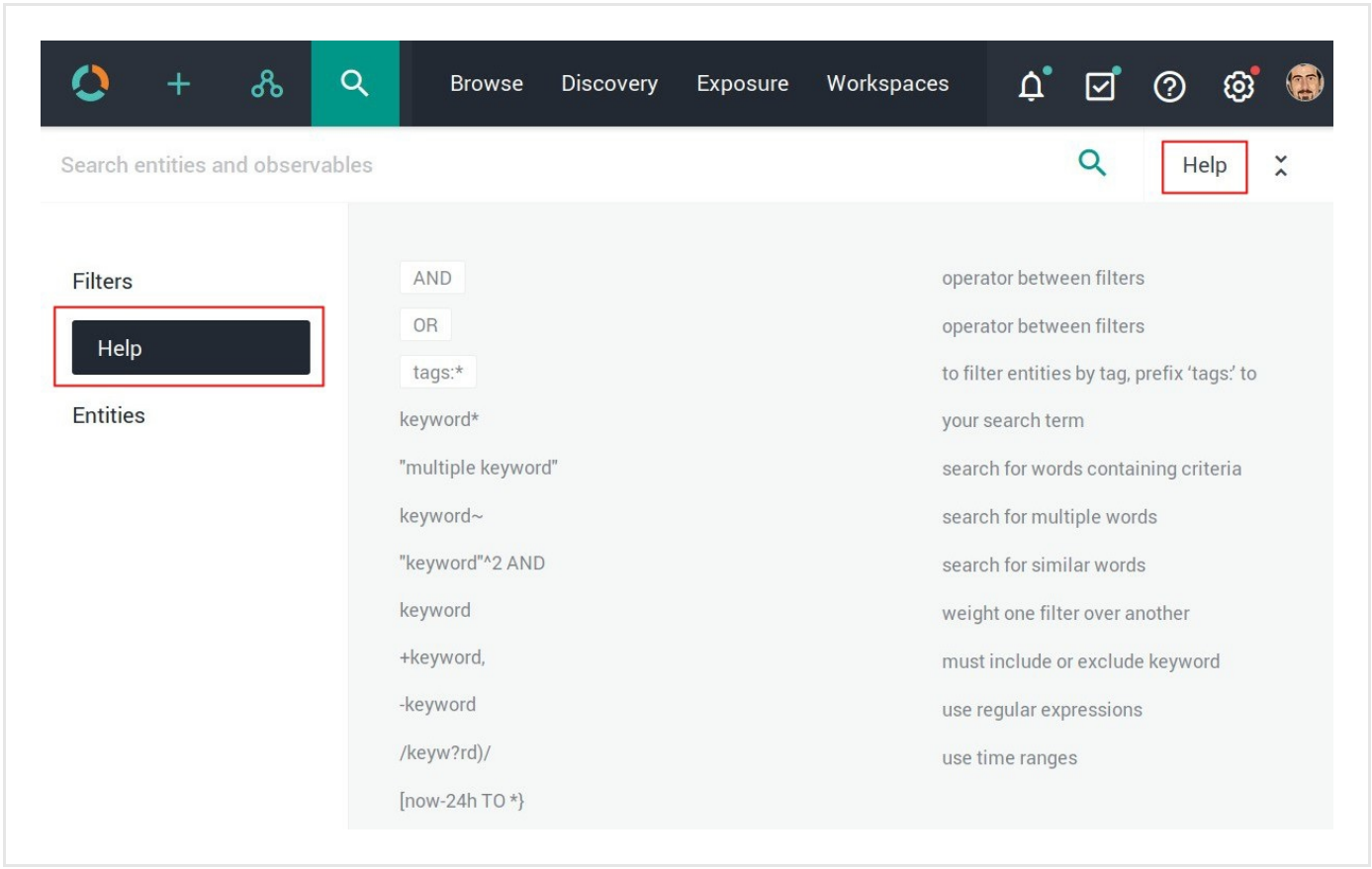
> The search functionality uses **Elasticsearch query syntax** (https://www.elastic.co/guide/en/elasticsearch/reference/current/full-text-queries.html).

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.



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



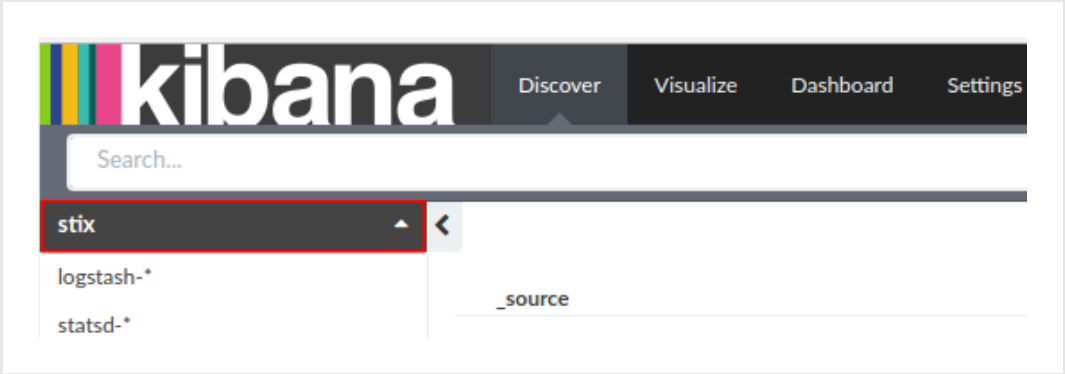
Use operators to combine multiple quick filters and create a more complex search query.
Example:

enrichment_extracts.kind:domain AND enrichment_extracts.meta.classification:high

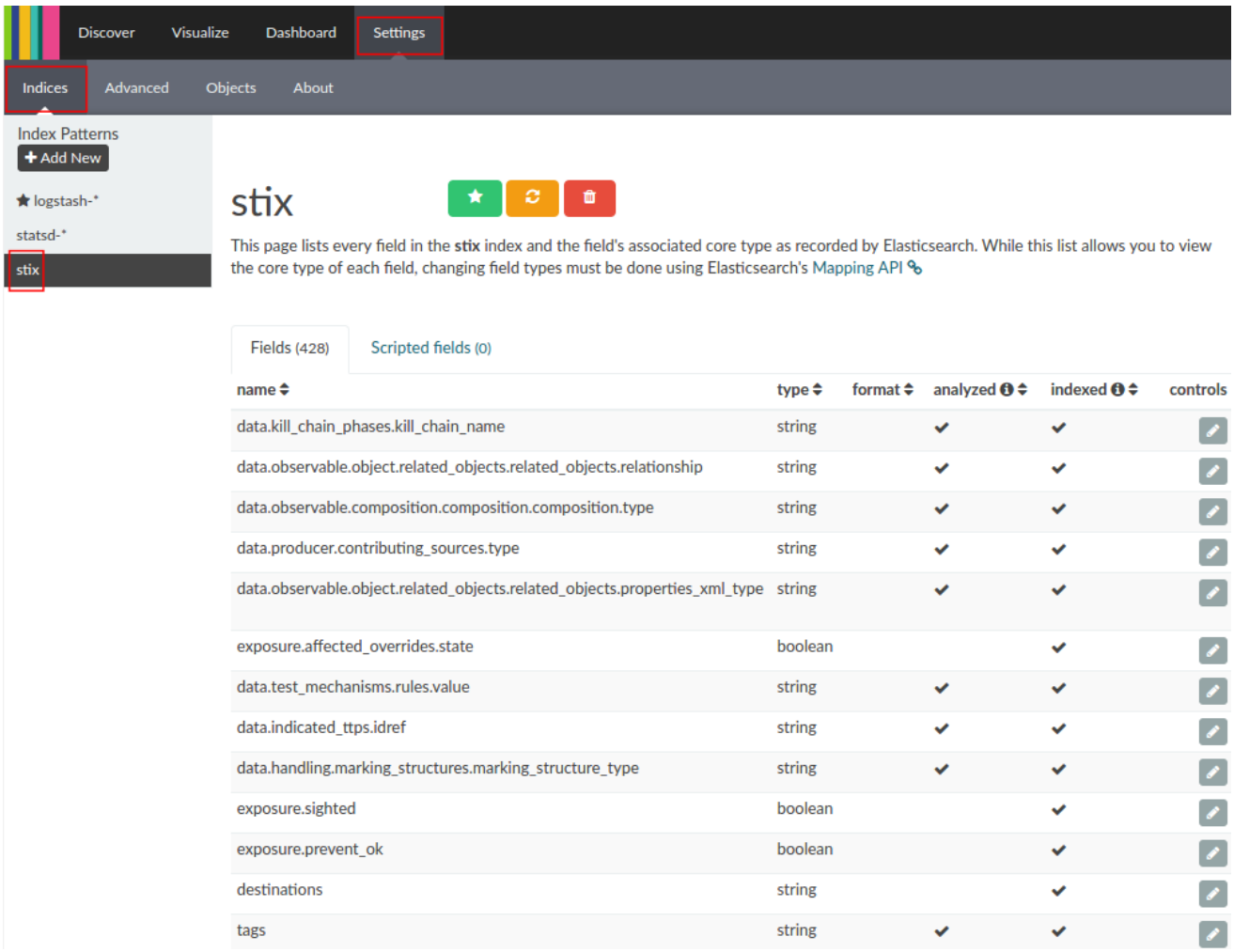
Field	Description	Example
enrichment_extracts.id	string — The alphanumeric ID string that uniquely identifies the enrichment observable.	`01h12x45-01q2-1234-od01-123456h78h90`
enrichment_extracts.kind	string — The enrichment observable data type.	`domain`
enrichment_extracts.meta.blacklisted	Boolean — An observable is blacklisted when it is included in the results returned by an *ignore* extraction rule. Allowed values: `true`, `false`.	`true`
enrichment_extracts.meta.classification	string — This value is defined in **Rules** by selecting appropriate options under **Action** and **Confidence**. Allowed classification metadata values are `good`, `bad`, and `unknown`.	`good`
enrichment_extracts.meta.confidence	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 `low`, `medium`, and `high`.	`high`
enrichment_extracts.value	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:

- Sign in to the platform with your user credentials.
- To access Kibana, in the web browser address bar enter a URL with the following format: `<platform_host>/api/kibana/app/kibana#/`.
  Keep the trailing `/`.
  Example: `https://platform.host.com/api/kibana/app/kibana#/`
- Select the **stix** index field:



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



Flashpoint integration

Integrate EclecticIQ Platform with Flashpoint AggregINT, Flashpoint Blueprint, and Flashpoint Thresher through the Flashpoint API.

Configure the enrichers

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 the enrichment information. Data sources can be incoming feeds, as well as other enrichers.
- Within the selected platform data sources, the entity type(s) to augment with the enrichment information.
- The enrichers to use to fetch the 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.

Observables

Observables augment the entities they are related to by providing additional context that can help discover indirect relationships or spawn new relationships between entities.
Observables are atomic and factual: an observable represents one discrete piece of information that describes a fact. For example, an IP address, a hash value, the name of a location or an actor.

The Flashpoint AggregINT, Flashpoint Blueprint, and Flashpoint Thresher enrichers share almost identical configuration options, the only differences being the number and the type of available Flashpoint datasets per enricher.

Configure enricher tasks

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

- On the top navigation bar click **✚ > Data management > Dataset > Enrichment** .

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Data management**.
- On the left-hand navigation sidebar click **Enrichment**.
- Click the enricher you want to configure or modify.
- On the enricher detail page, click the **Edit** button.

> ✔ On the forms, 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 content of the outgoing feed with a predefined reliability value to help other users assess how trustworthy the feed 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 rule is executed automatically. If you deselect it, you need to run the rule manually.
- Under **Parameters**, define the specific configuration options for the selected enricher, where applicable.
- Click **Save** to store your changes, or **Cancel** to discard them.

Configure enricher rules

Add enricher rules

To add a new enricher rule, do the following:

- On the top navigation bar click **✚ > Rules > Enrichment**.

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.

- On the left-hand navigation sidebar click **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 **+ Rule** button.

> ✔ On the forms, input fields marked with an asterisk are required.

On the **Rules > Enrichment > Create** 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 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 creation work.

Edit enricher rules

To edit enricher rules, do the following:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.

- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 detail pane, click **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**.

> ✔ On the forms, 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 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.
- Click **Delete** on the rule detail pane.

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.

Run the enricher

Automatically

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

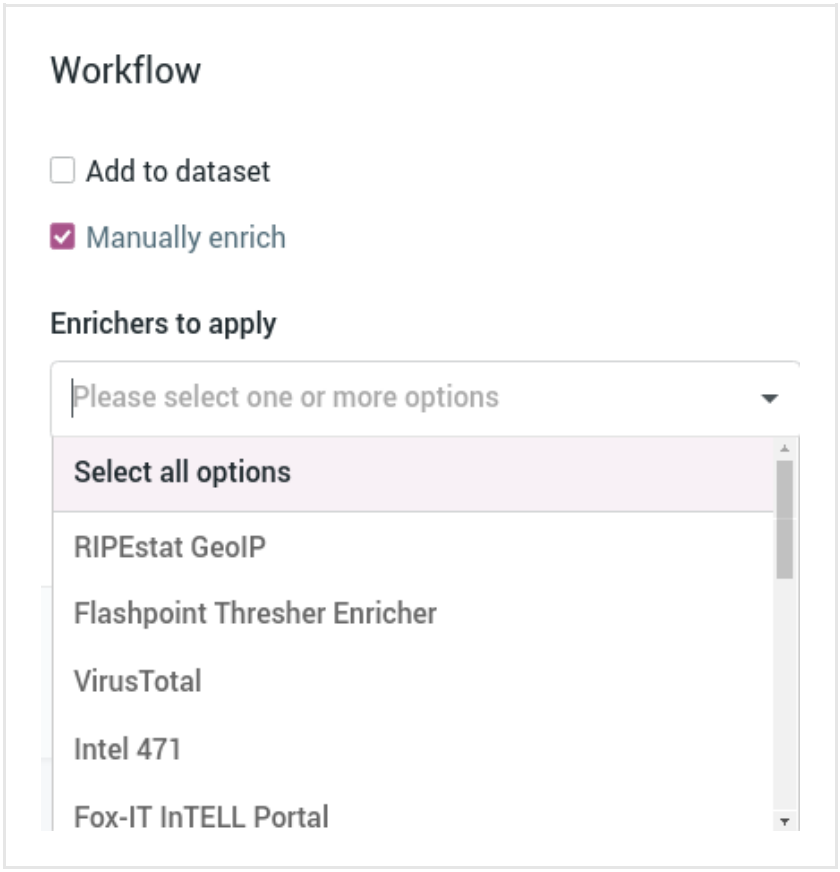
Rules give you control over the type of information you want to retrieve or exclude, and what you want to do with it. You can assign one or more enricher sources to specific observable types. You can set multiple filters to cover usage scenarios as needed. You can then examine the returned enrichment 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**.

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.

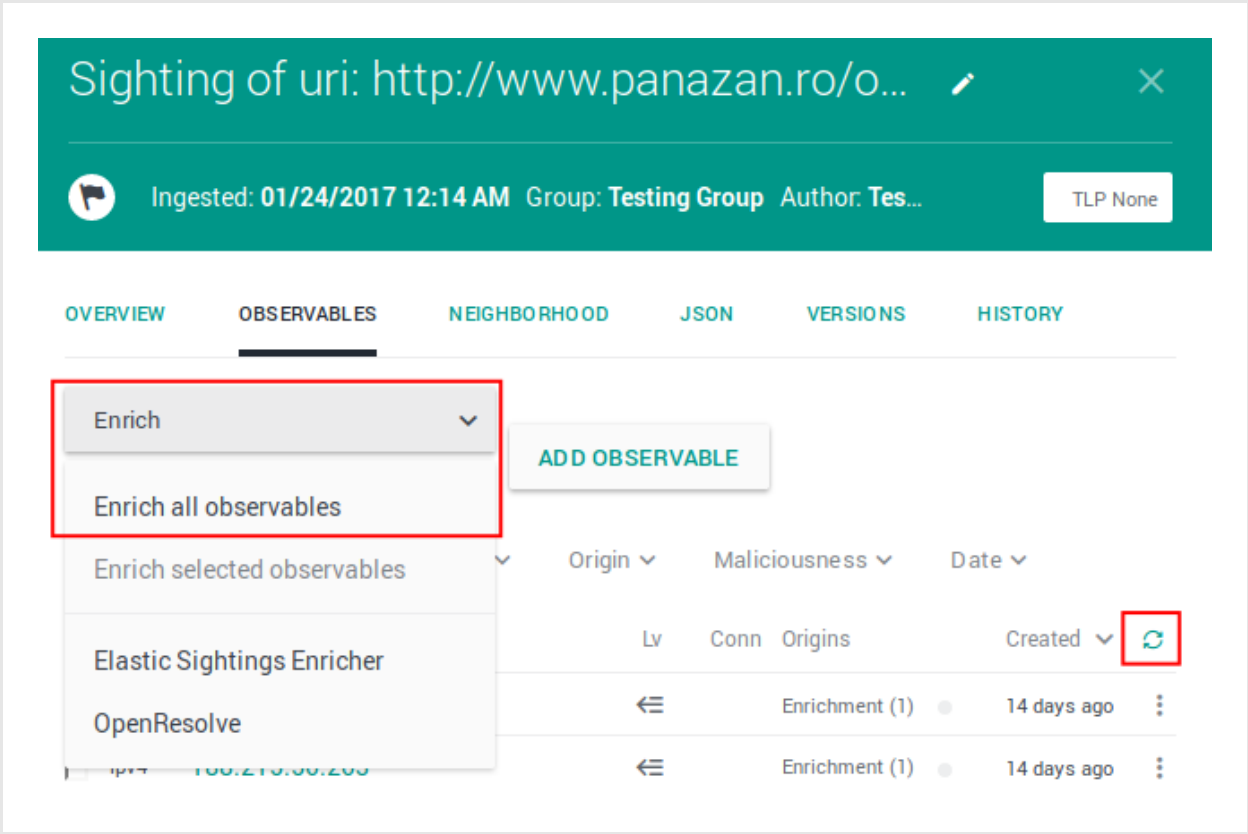
- On the entity detail pane, click **Observables**.
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.

To manually enrich the entity observables:

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

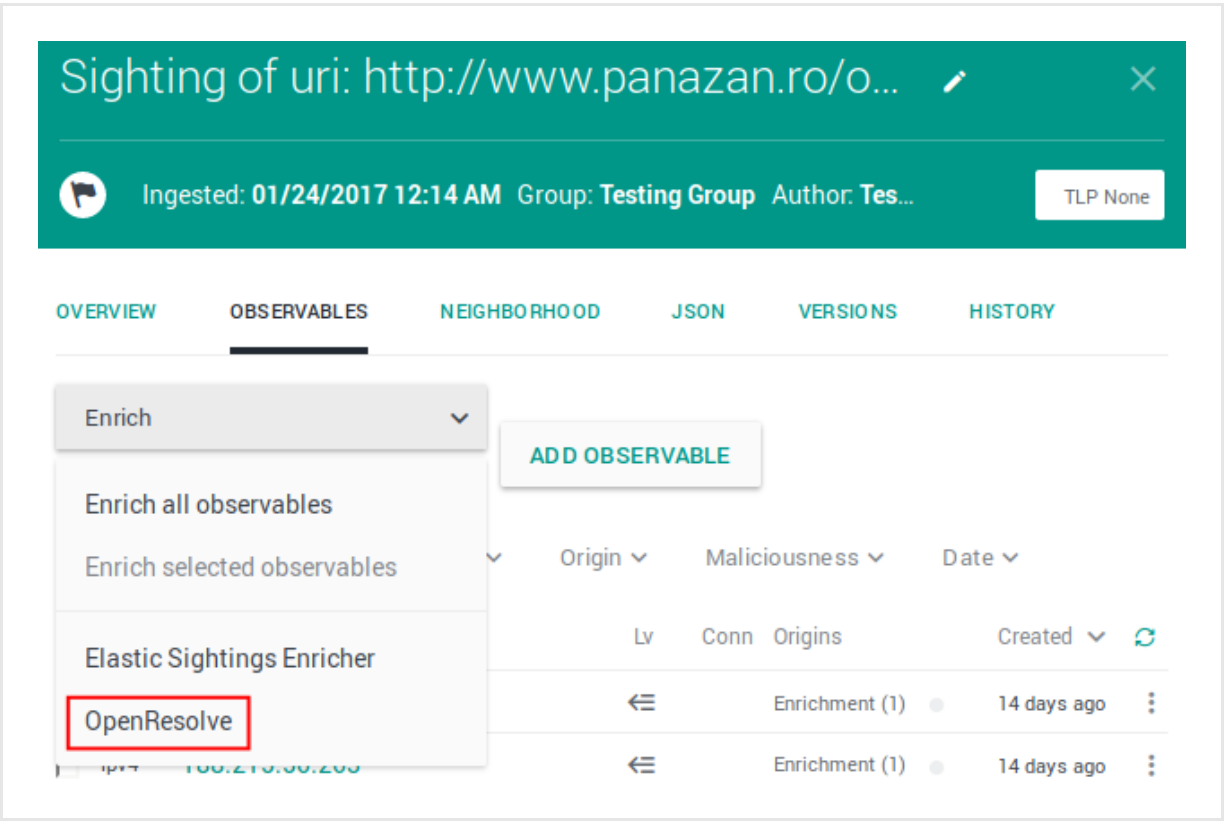
Alternatively:

- From the **Enrich** drop-down menu, select **Enrich all observables**.
- 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:

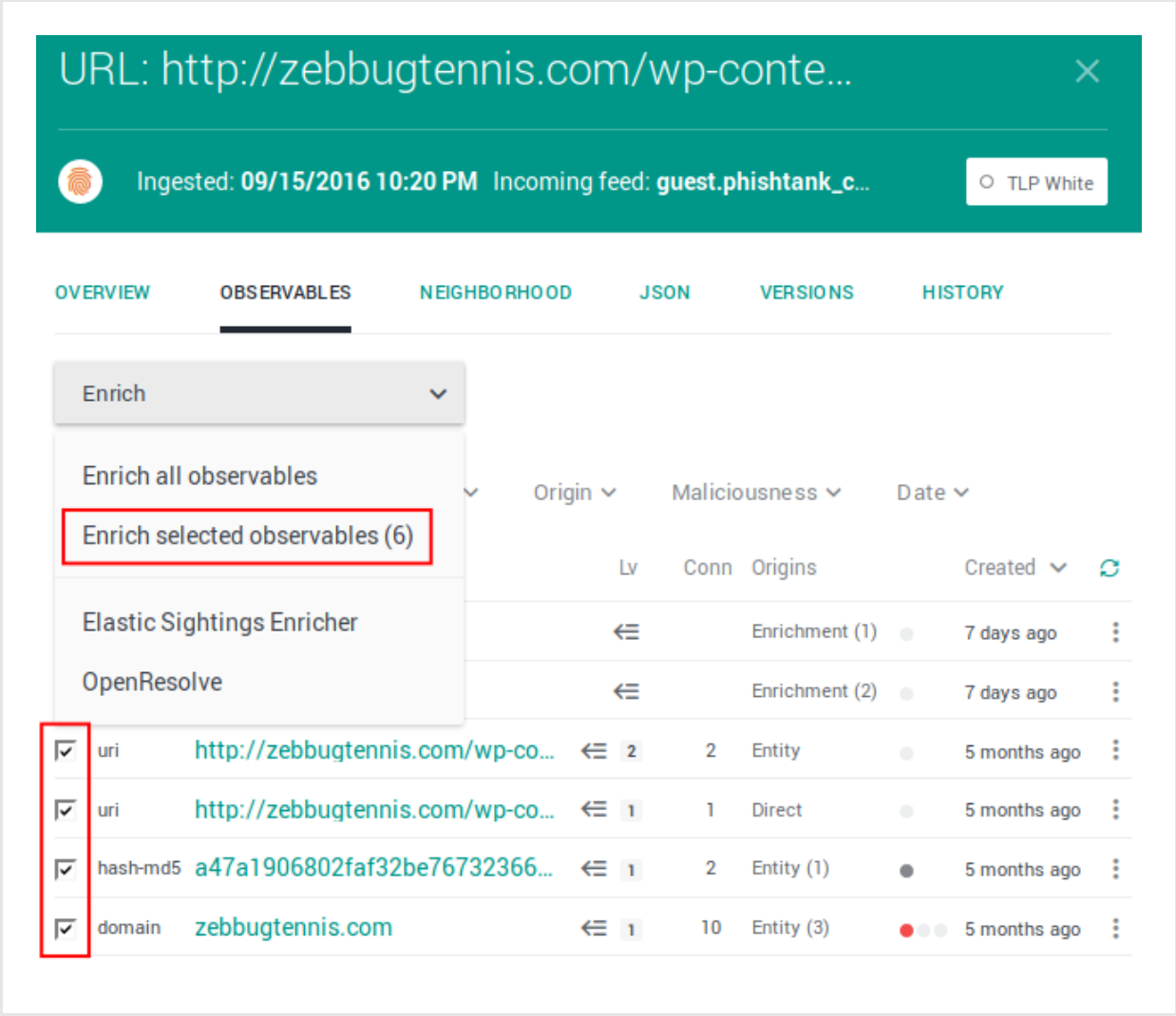
- Select it from the **Enrich** drop-down menu, and then click it.
- The platform polls the specified enricher for the entity, and it enriches all the entity observables with the retrieved data.



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 selected observables**.
- The platform polls all applicable enrichers for the entity, and it enriches the selected entity observables with the retrieved data.



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

Flashpoint enrichers can take the following observable types as input:

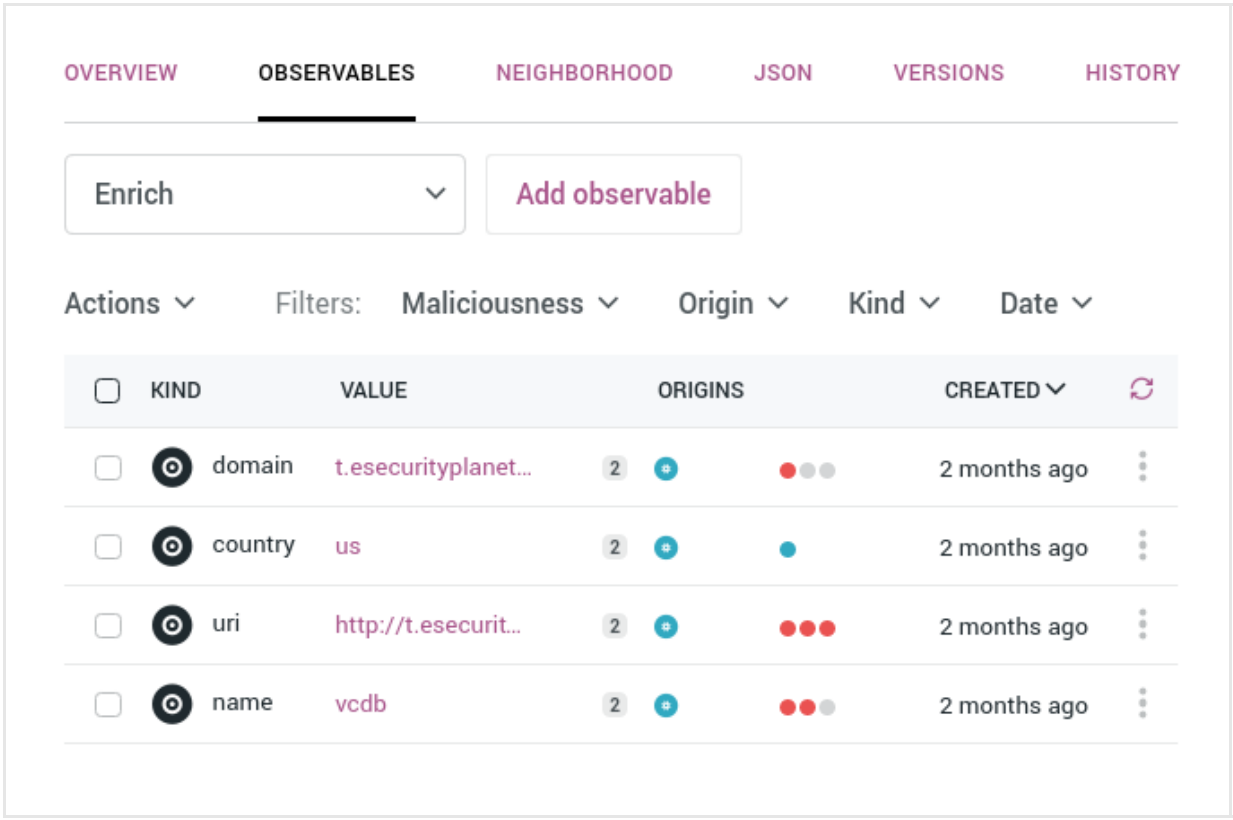
- *ipv4, ipv6, domain, host, uri, email, actor-id, hash-md5, hash-sha1, hash-sha256, hash-sha512*

The enrichers use these data types to look for additional information on observables. Any entity types supporting these observable types can be enriched with Flashpoint enrichers.

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**.

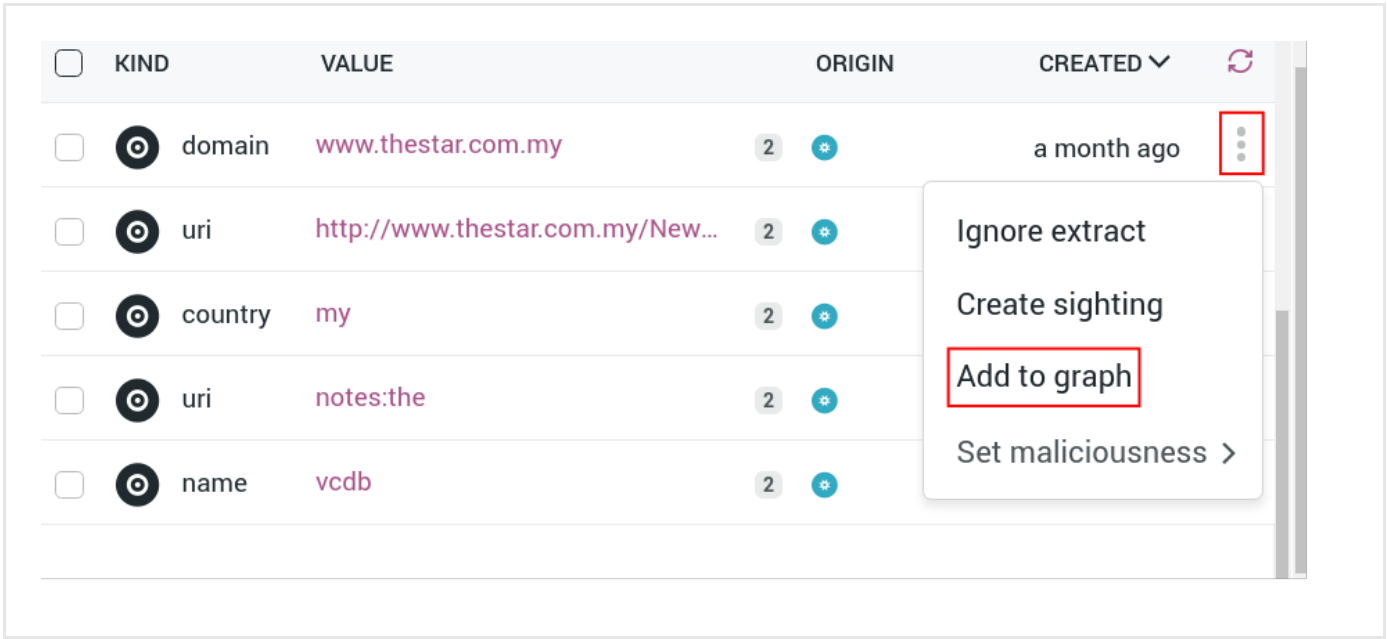
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.



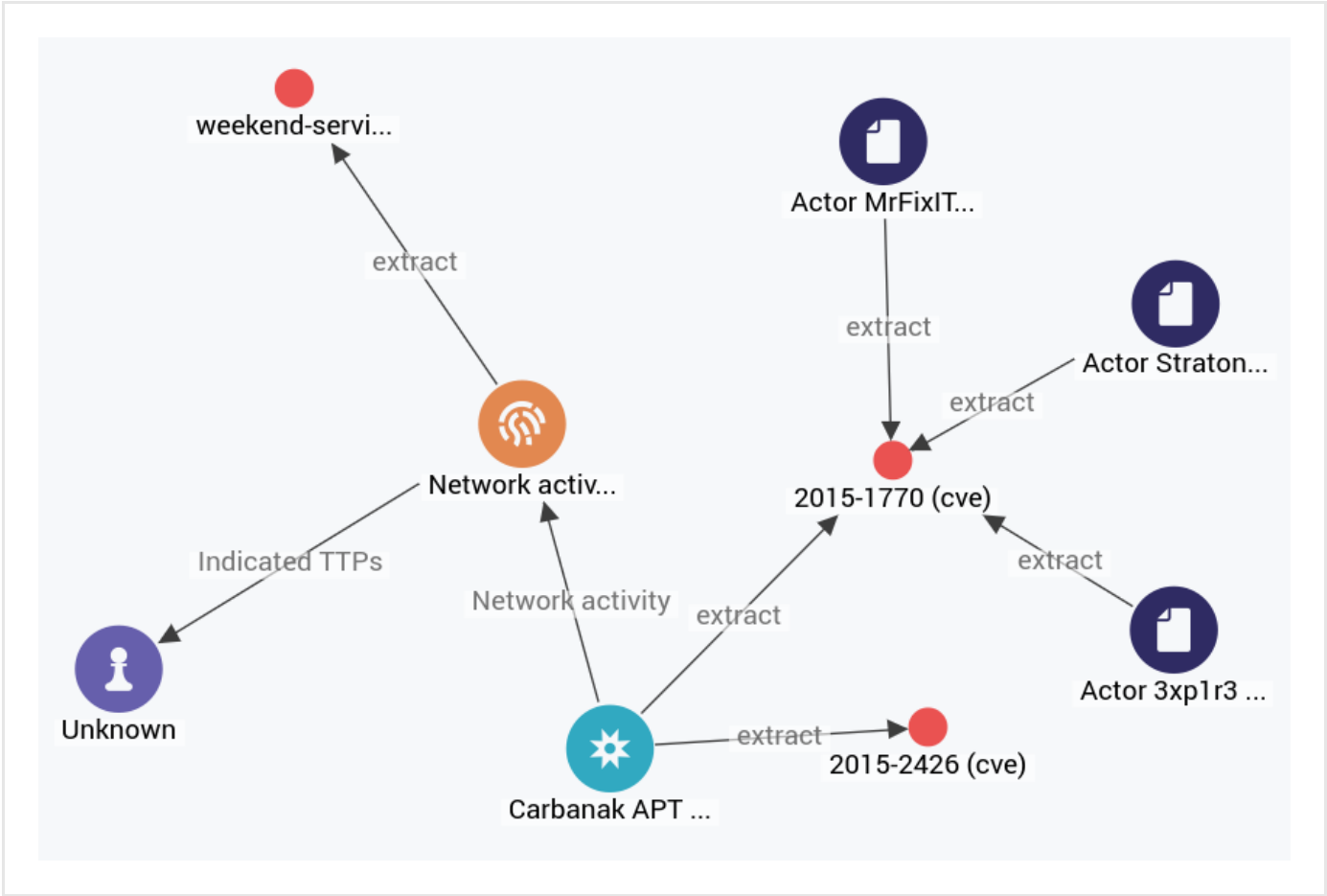
Review enrichment observables on the graph

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

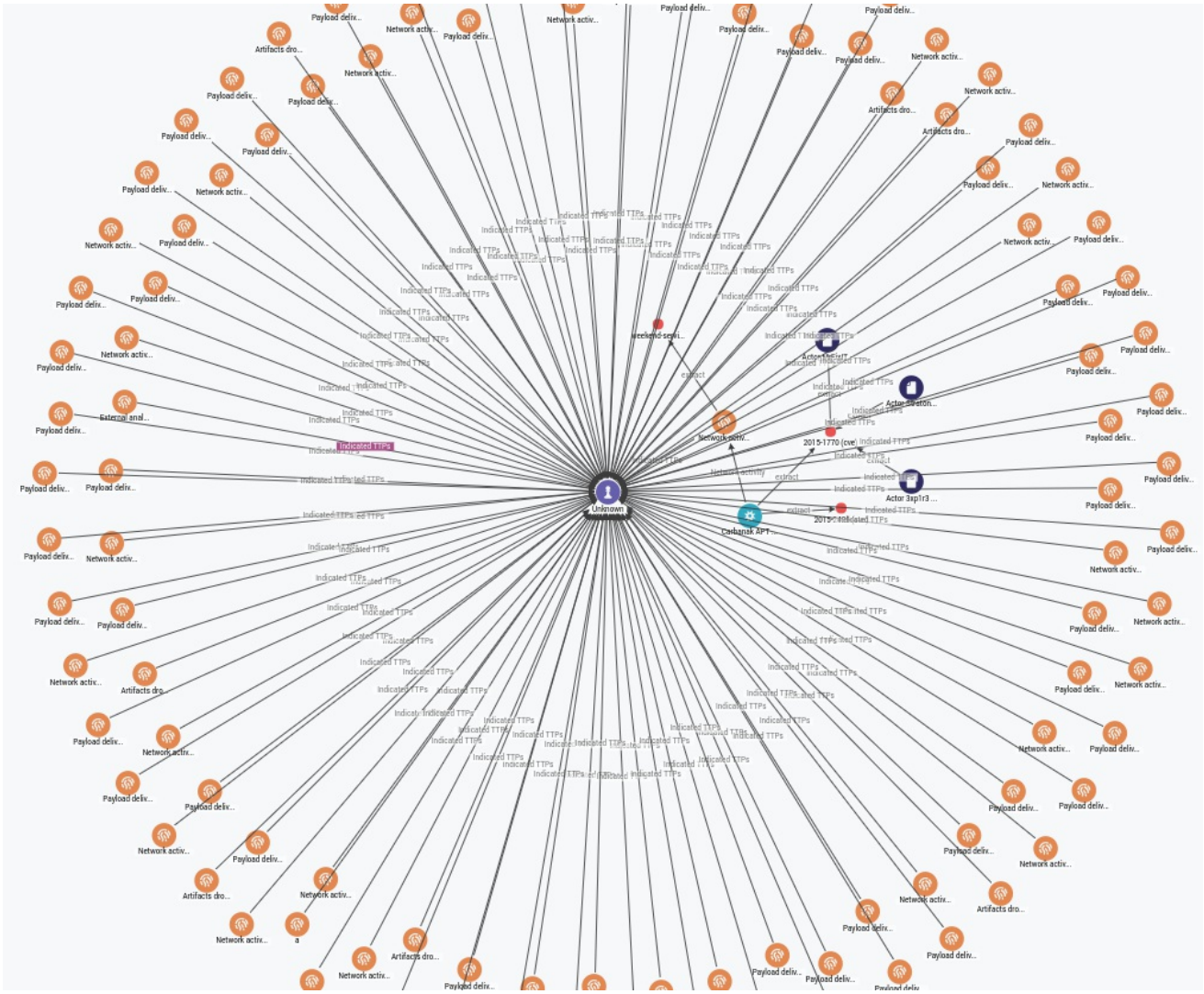
- On the row corresponding to the observable you want to load onto the graph, click the ⋮ icon, and then select **Add to graph**.



- 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** .

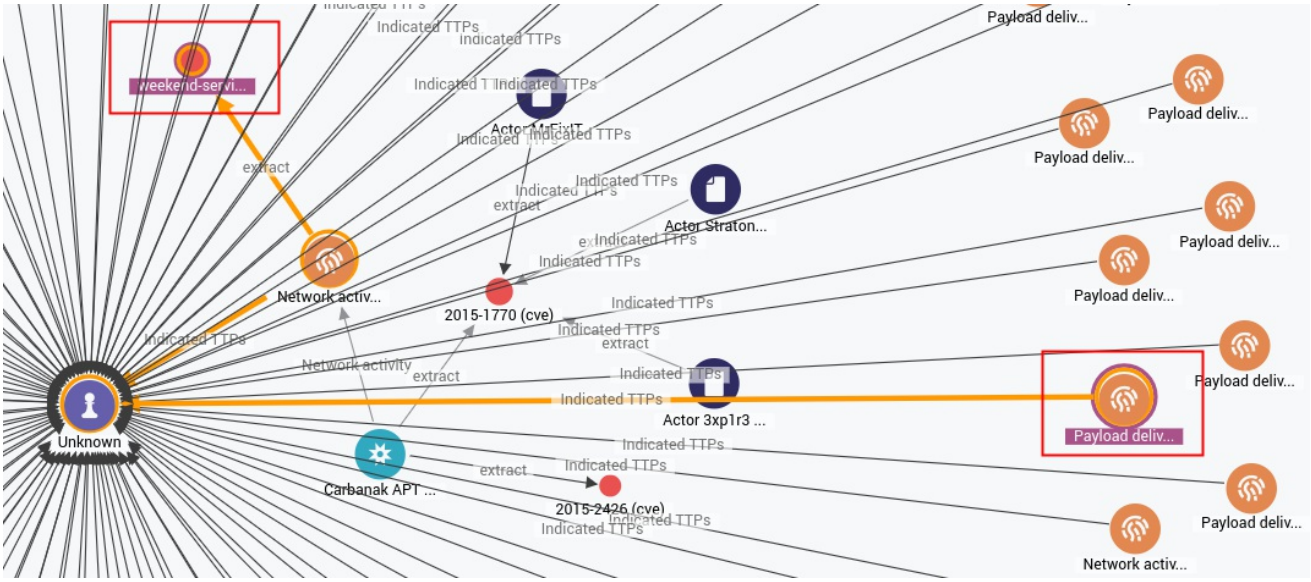


- Right-click an extract or an entity for further inspection and from the context menu select **Load entities > All**, **Load observables > All** or **Load entities by extract > All** .



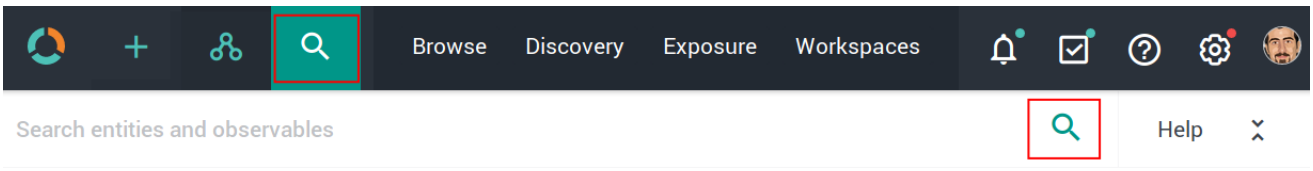
To see how entities, observables and enrichment observables are connected, and to inspect relationships between distant items, do the following:

- **CTRL** + click two nodes on the graph to select them.
- Right-click either selected node, and from the context menu select **Find path** to query the graph database about the existence of a path between the nodes, or **Show path** to highlight asn existing path on the graph.
- If a path does exist, the selected nodes and all the intermediate ones are highlighted on the graph to show the path that links them.



Search for enrichment observables

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

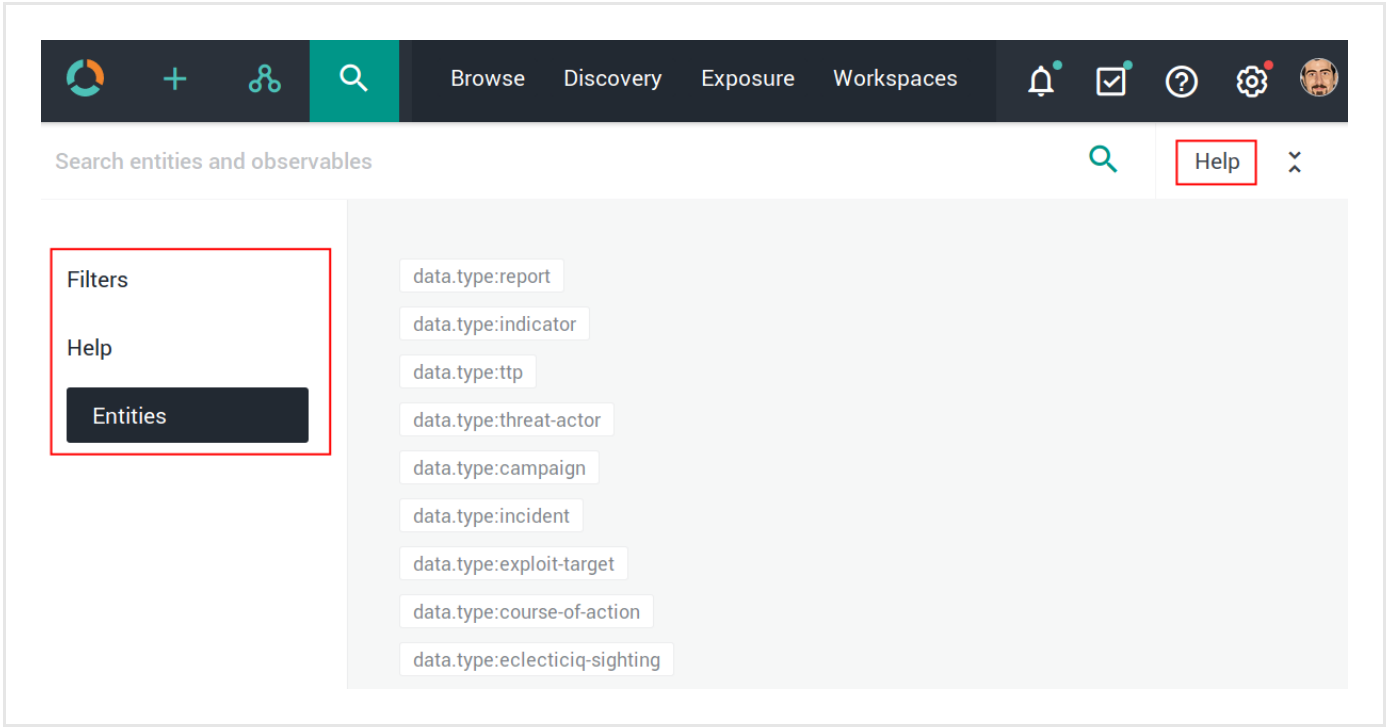


Enter search terms and search queries, and then press **ENTER** or click the search icon to run the search. Searches you run through this search box are executed platform-wide.

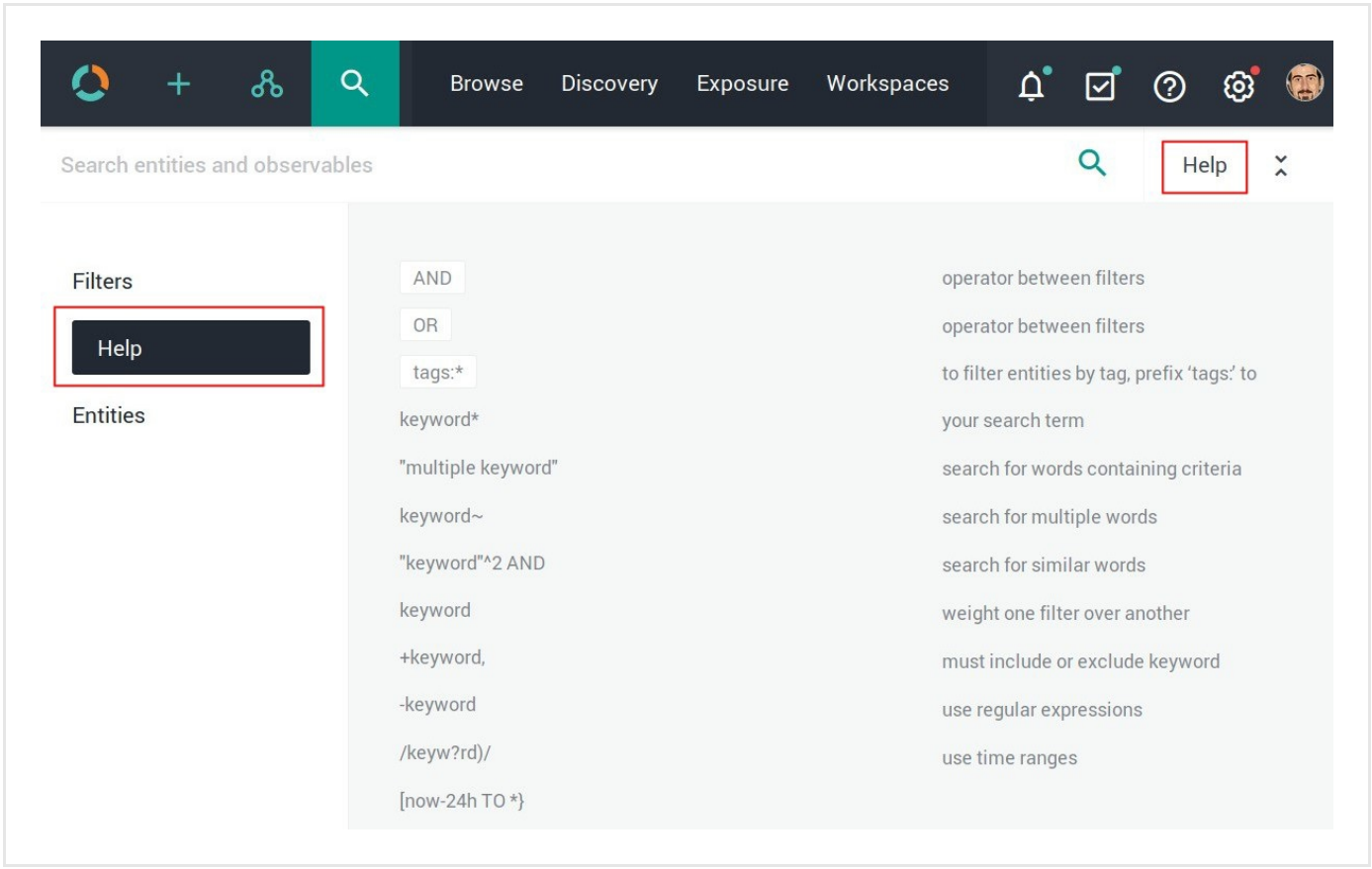
> The search functionality uses **Elasticsearch query syntax** (https://www.elastic.co/guide/en/elasticsearch/reference/current/full-text-queries.html).

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.



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



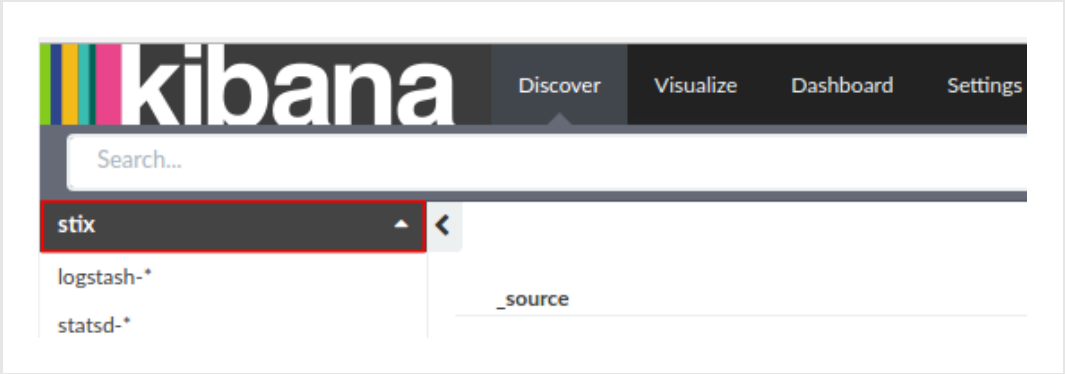
Use operators to combine multiple quick filters and create a more complex search query.
Example:

enrichment_extracts.kind:domain AND enrichment_extracts.meta.classification:high

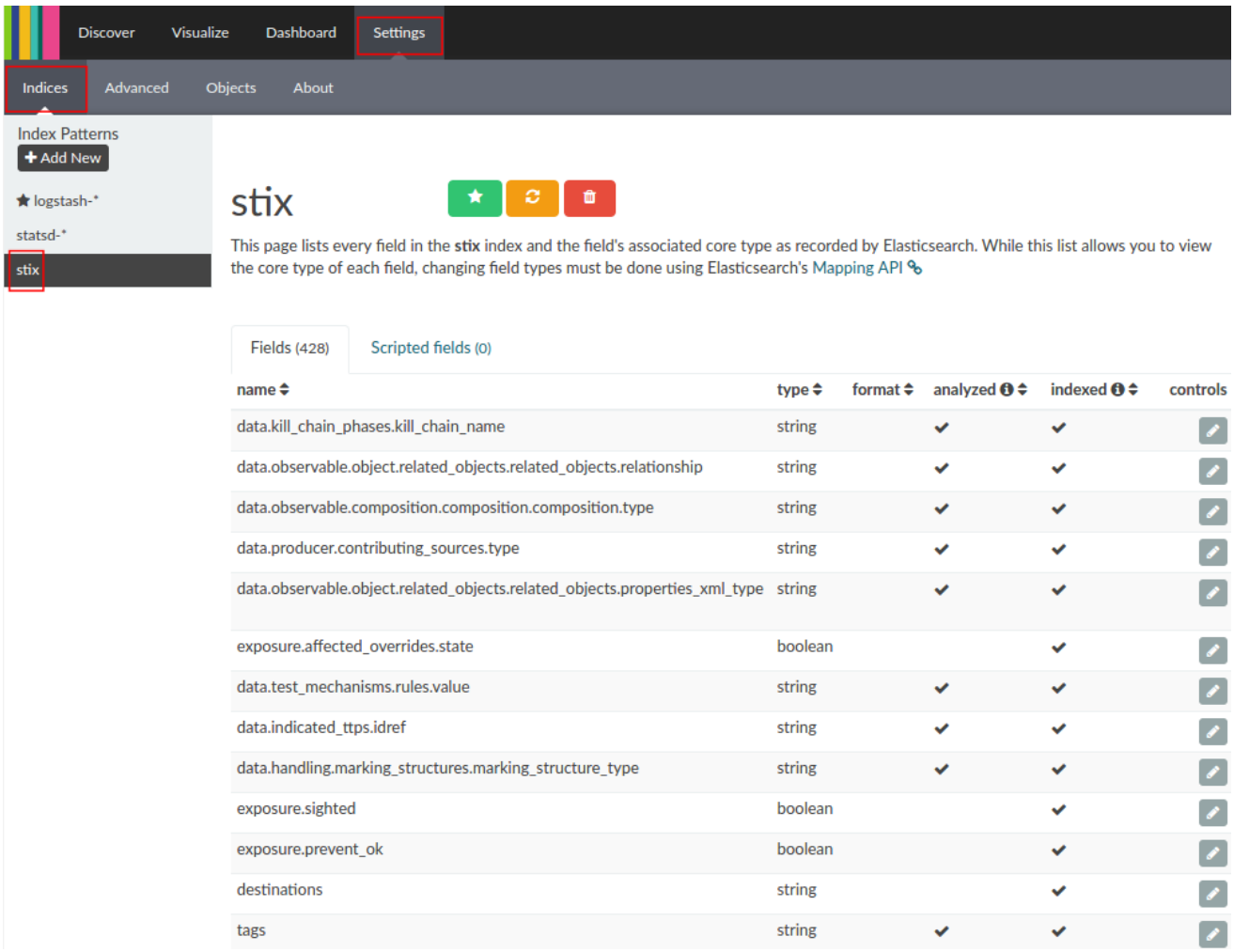
Field	Description	Example
enrichment_extracts.id	string — The alphanumeric ID string that uniquely identifies the enrichment observable.	`01h12x45-01q2-1234-od01-123456h78h90`
enrichment_extracts.kind	string — The enrichment observable data type.	`domain`
enrichment_extracts.meta.blacklisted	Boolean — An observable is blacklisted when it is included in the results returned by an *ignore* extraction rule. Allowed values: `true`, `false`.	`true`
enrichment_extracts.meta.classification	string — This value is defined in **Rules** by selecting appropriate options under **Action** and **Confidence**. Allowed classification metadata values are `good`, `bad`, and `unknown`.	`good`
enrichment_extracts.meta.confidence	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 `low`, `medium`, and `high`.	`high`
enrichment_extracts.value	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:

- Sign in to the platform with your user credentials.
- To access Kibana, in the web browser address bar enter a URL with the following format: `<platform_host>/api/kibana/app/kibana#/`.
  Keep the trailing `/`.
  Example: `https://platform.host.com/api/kibana/app/kibana#/`
- Select the **stix** index field:



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



PassiveTotal integration

Integrate EclecticIQ Platform with RiskIQ PassiveTotal to retrieve active/passive DNS, IP, domain, and malware information.

Configure the enrichers

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 the enrichment information. Data sources can be incoming feeds, as well as other enrichers.
- Within the selected platform data sources, the entity type(s) to augment with the enrichment information.
- The enrichers to use to fetch the 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.

Observables

Observables augment the entities they are related to by providing additional context that can help discover indirect relationships or spawn new relationships between entities.
Observables are atomic and factual: an observable represents one discrete piece of information that describes a fact. For example, an IP address, a hash value, the name of a location or an actor.

The PassiveTotal Whois, PassiveTotal Passive DNS, PassiveTotal IP/Domain, and PassiveTotal Malware enrichers share almost identical configuration options, the only differences being the available PassiveTotal dataset types per enricher.

Configure enricher tasks

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

- On the top navigation bar click **+ > Data management > Dataset > Enrichment** .

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Data management**.
- On the left-hand navigation sidebar click **Enrichment**.
- Click the enricher you want to configure or modify.
- On the enricher detail page, click the **Edit** button.

> ✔ On the forms, 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 content of the outgoing feed with a predefined reliability value to help other users assess how trustworthy the feed 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 rule is executed automatically. If you deselect it, you need to run the rule manually.
- Under **Parameters**, define the specific configuration options for the selected enricher, where applicable.
- Click **Save** to store your changes, or **Cancel** to discard them.

Configure enricher rules

Add enricher rules

To add a new enricher rule, do the following:

- On the top navigation bar click **✚ > Rules > Enrichment**.

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.

- On the left-hand navigation sidebar click **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 **+ Rule** button.

> ✔ On the forms, input fields marked with an asterisk are required.

On the **Rules > Enrichment > Create** 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 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 creation work.

Edit enricher rules

To edit enricher rules, do the following:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.

- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 detail pane, click **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**.

> ✔ On the forms, 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 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.
- Click **Delete** on the rule detail pane.

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.

Run the enricher

Automatically

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

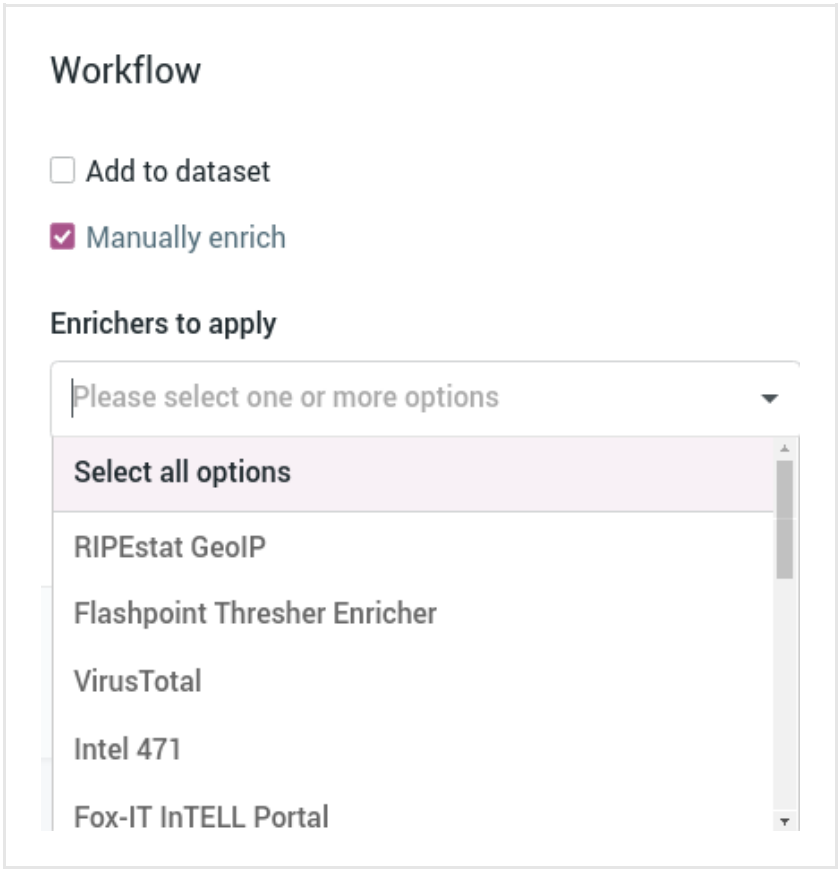
Rules give you control over the type of information you want to retrieve or exclude, and what you want to do with it. You can assign one or more enricher sources to specific observable types. You can set multiple filters to cover usage scenarios as needed. You can then examine the returned enrichment 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**.

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.

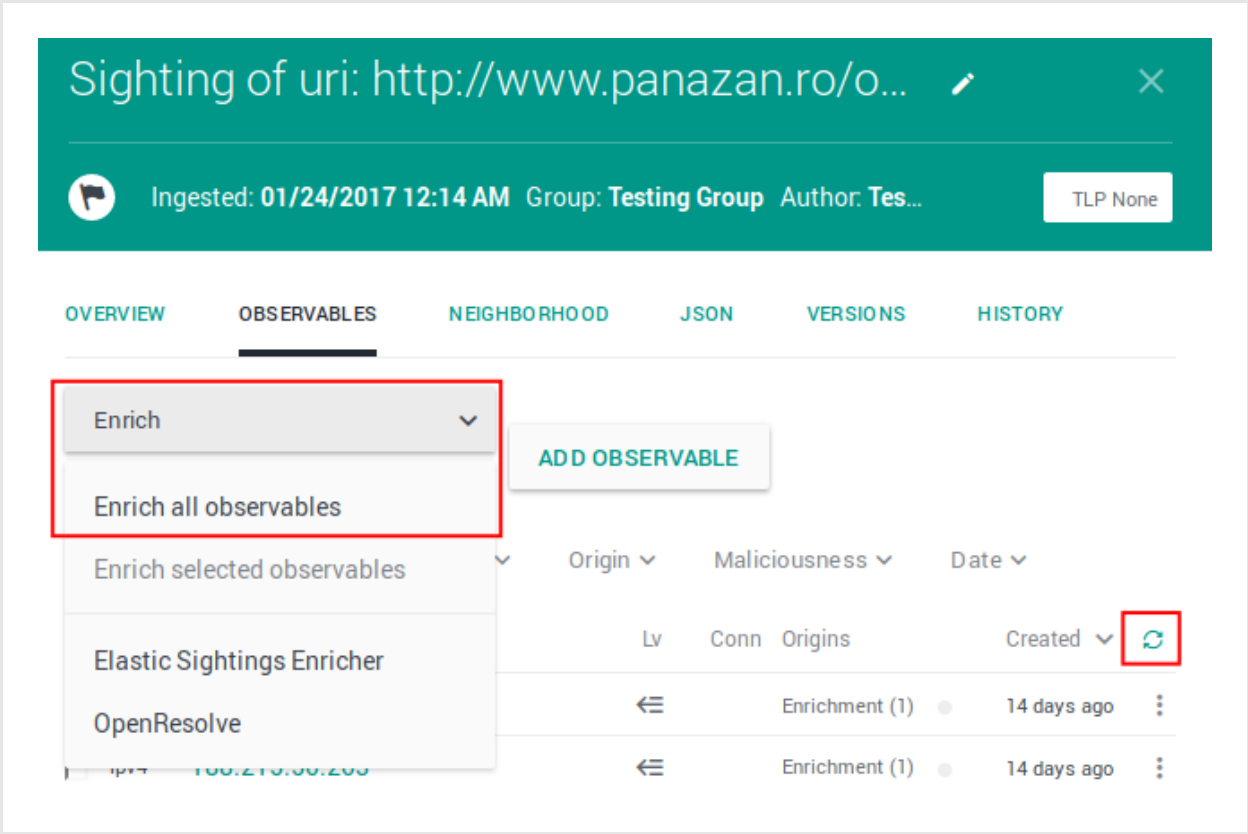
- On the entity detail pane, click **Observables**.
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.

To manually enrich the entity observables:

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

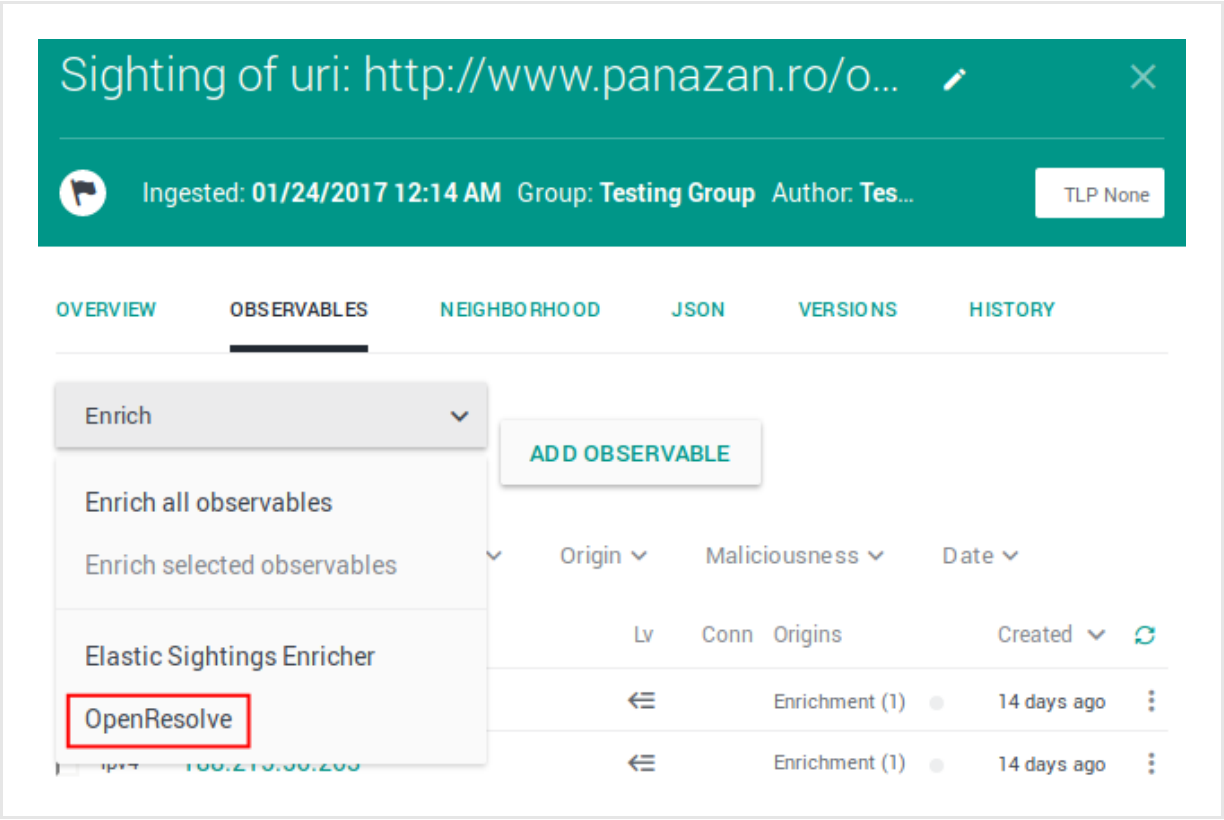
Alternatively:

- From the **Enrich** drop-down menu, select **Enrich all observables**.
- 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:

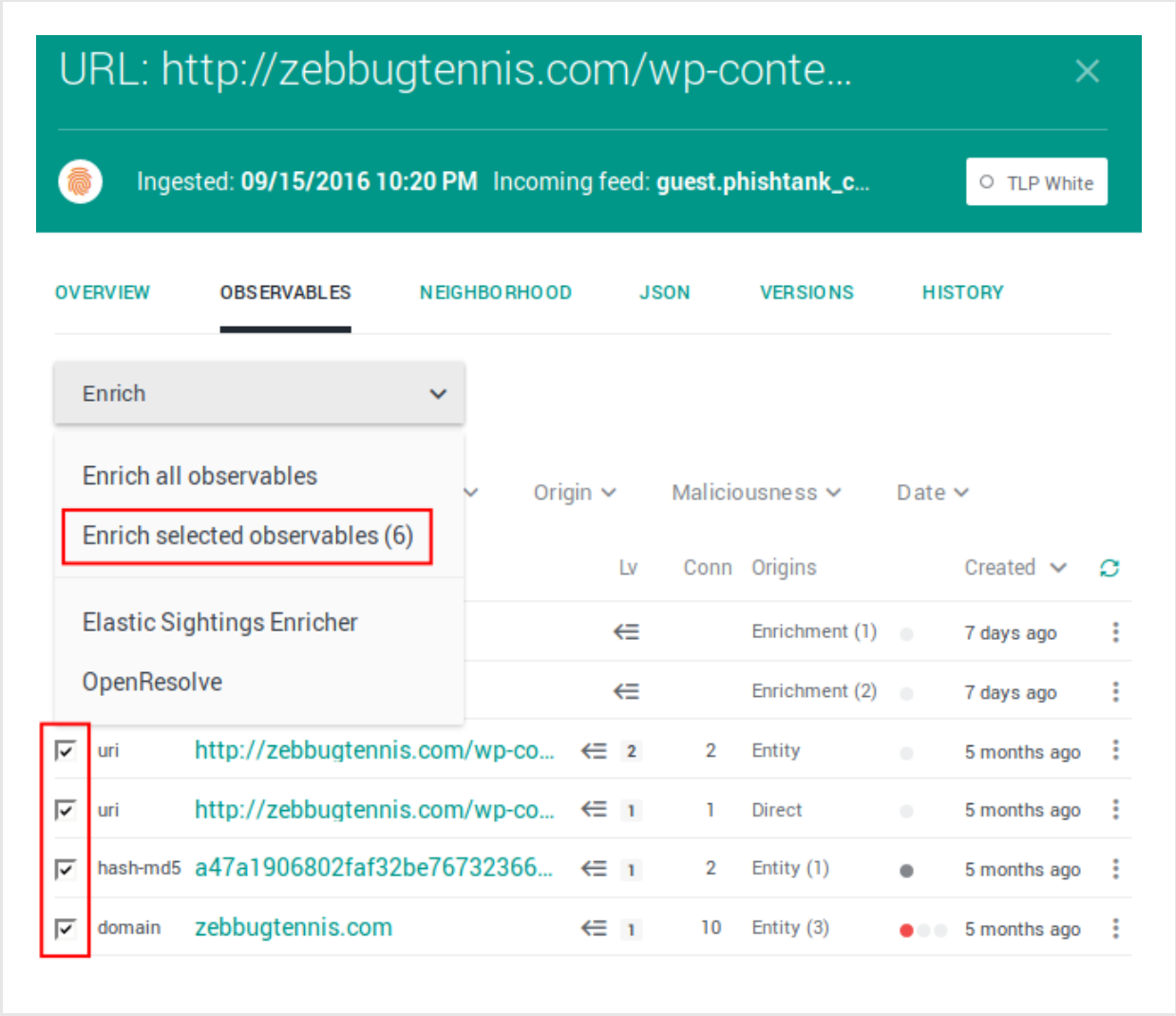
- Select it from the **Enrich** drop-down menu, and then click it.
- The platform polls the specified enricher for the entity, and it enriches all the entity observables with the retrieved data.



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 selected observables**.
- The platform polls all applicable enrichers for the entity, and it enriches the selected entity observables with the retrieved data.



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

RiskIQ PassiveTotal enrichers can take the following observable types as input:

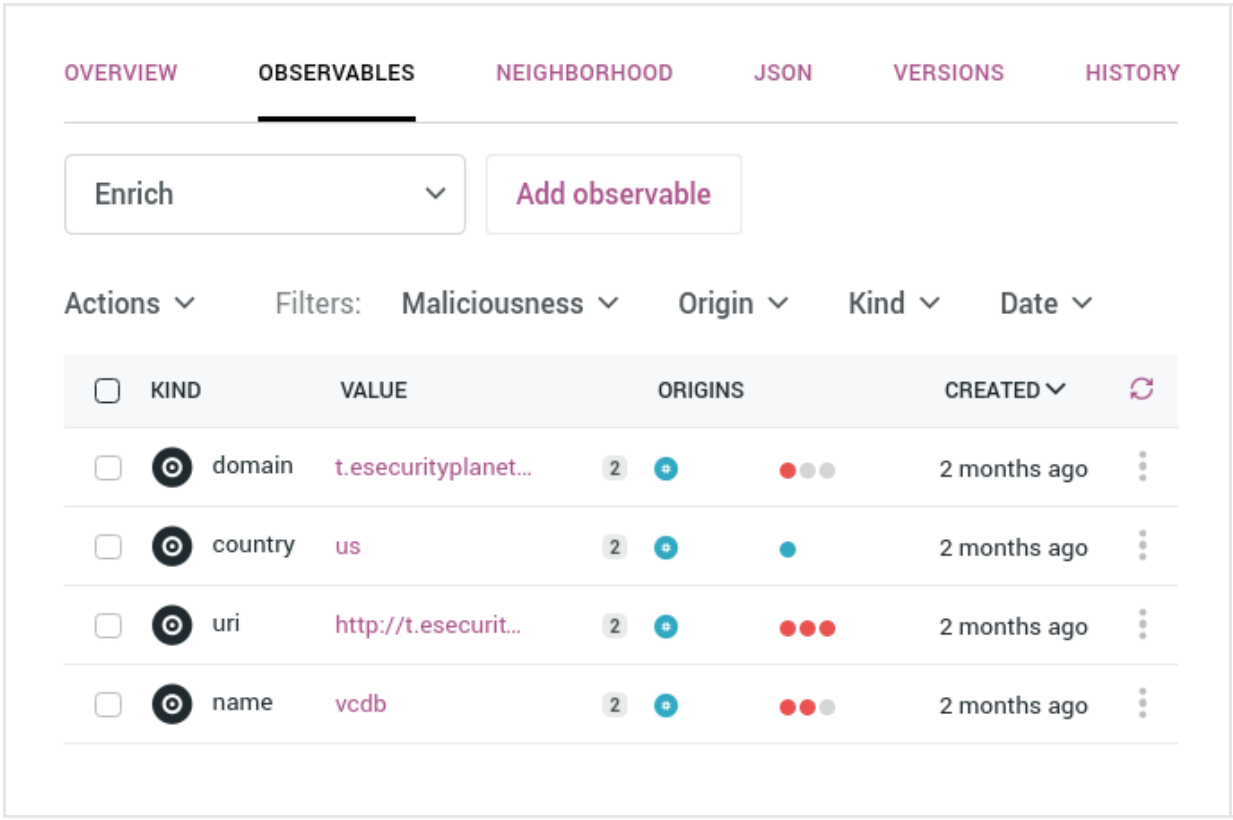
- *ipv4, ipv6, domain, host*

RiskIQ PassiveTotal enrichers use these data types to look for additional information on observables. Any entity types supporting these observable types can be enriched with RiskIQ PassiveTotal enrichers.

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**.

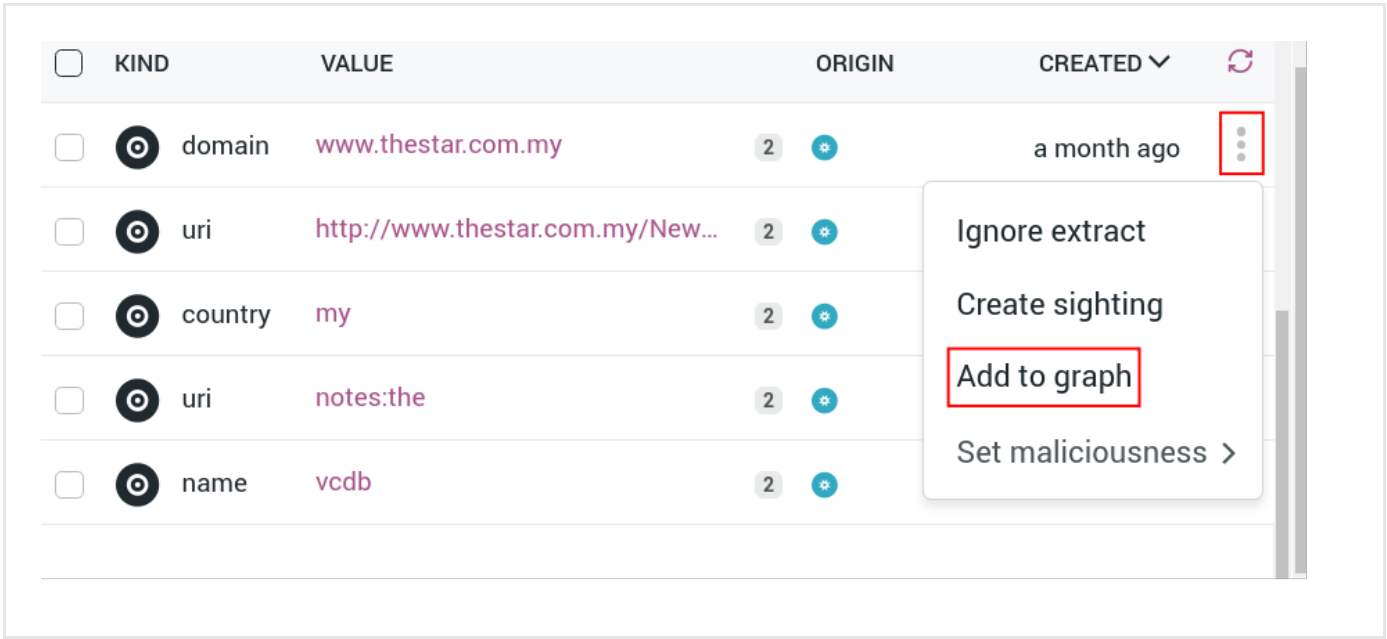
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.



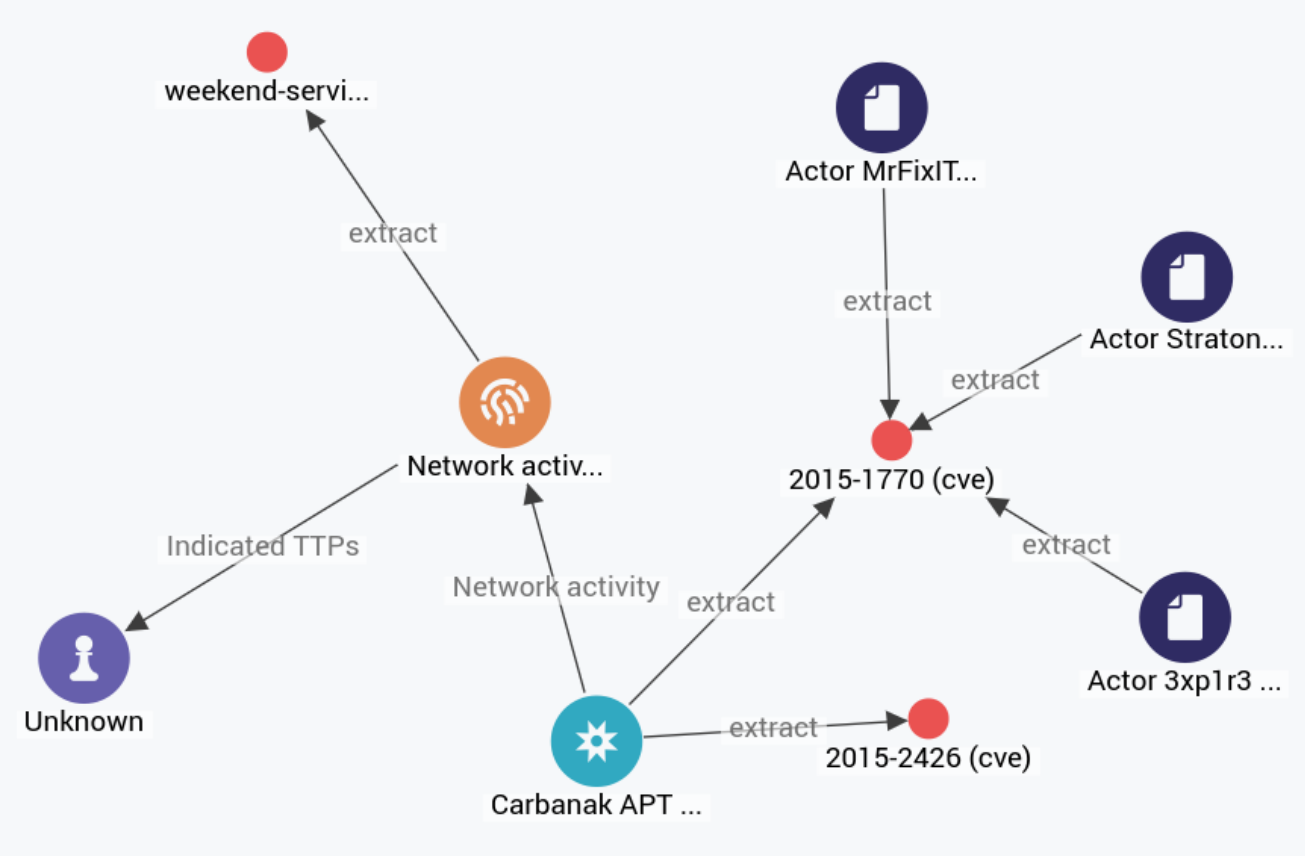
Review enrichment observables on the graph

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

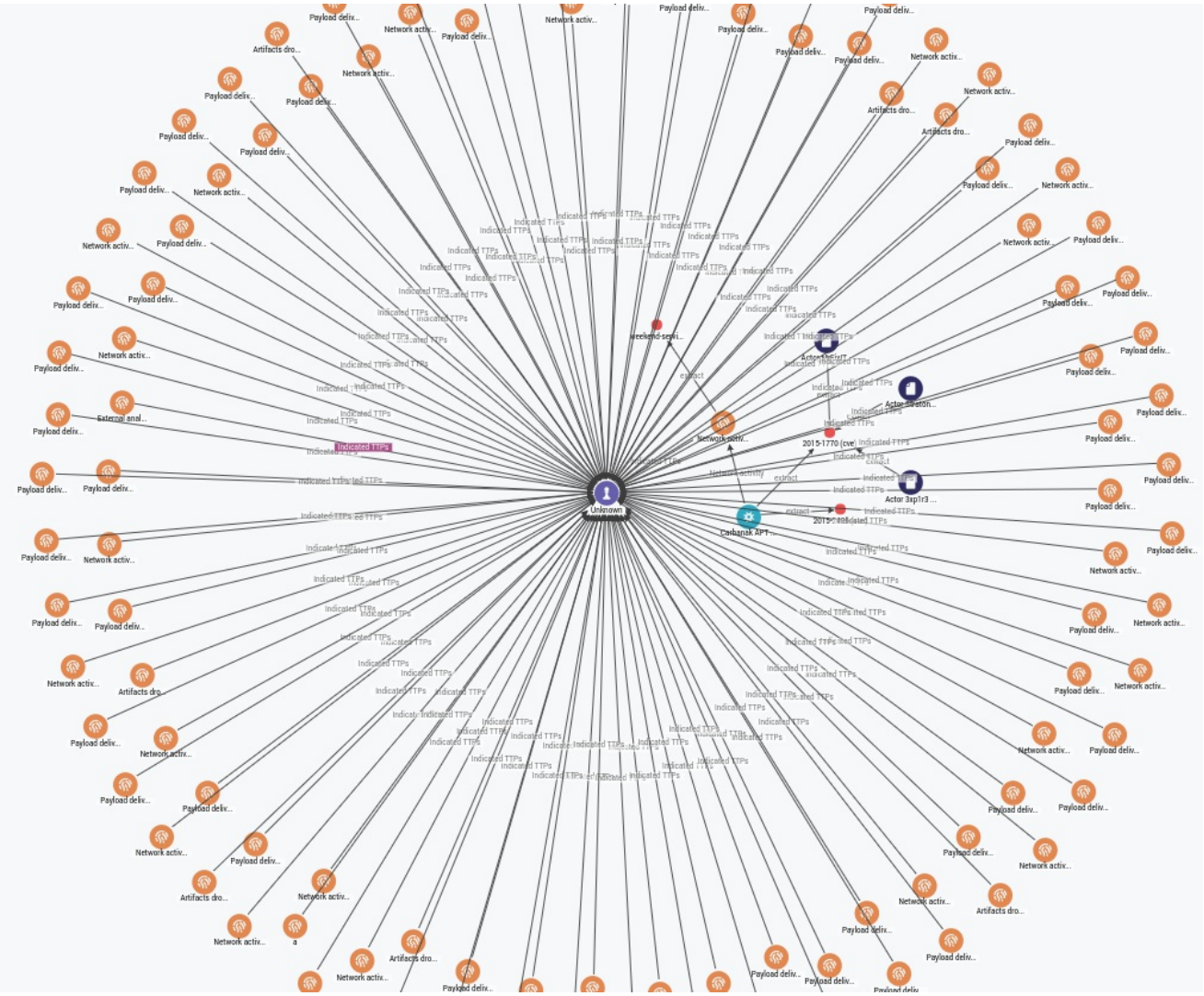
- On the row corresponding to the observable you want to load onto the graph, click the ⋮ icon, and then select **Add to graph**.



- 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** .

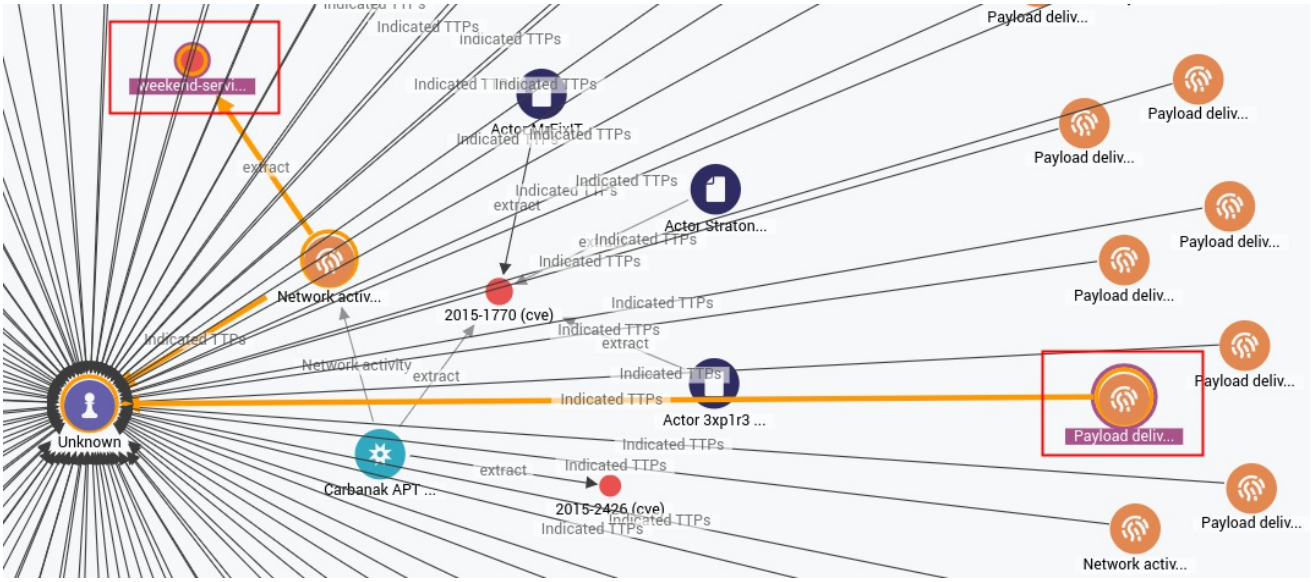


- Right-click an extract or an entity for further inspection and from the context menu select **Load entities > All**, **Load observables > All** or **Load entities by extract > All** .



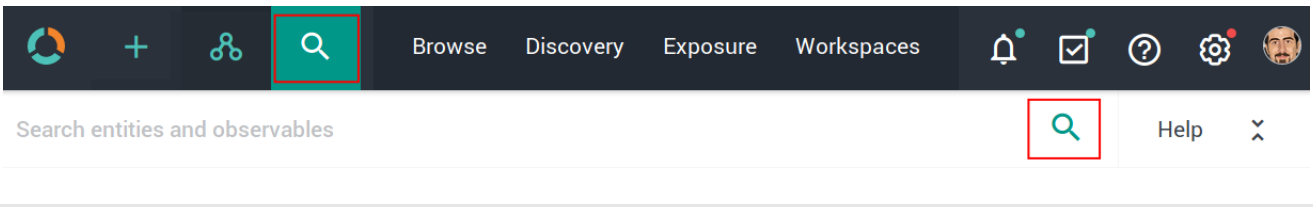
To see how entities, observables and enrichment observables are connected, and to inspect relationships between distant items, do the following:

- **CTRL** + click two nodes on the graph to select them.
- Right-click either selected node, and from the context menu select **Find path** to query the graph database about the existence of a path between the nodes, or **Show path** to highlight asn existing path on the graph.
- If a path does exist, the selected nodes and all the intermediate ones are highlighted on the graph to show the path that links them.



Search for enrichment observables

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

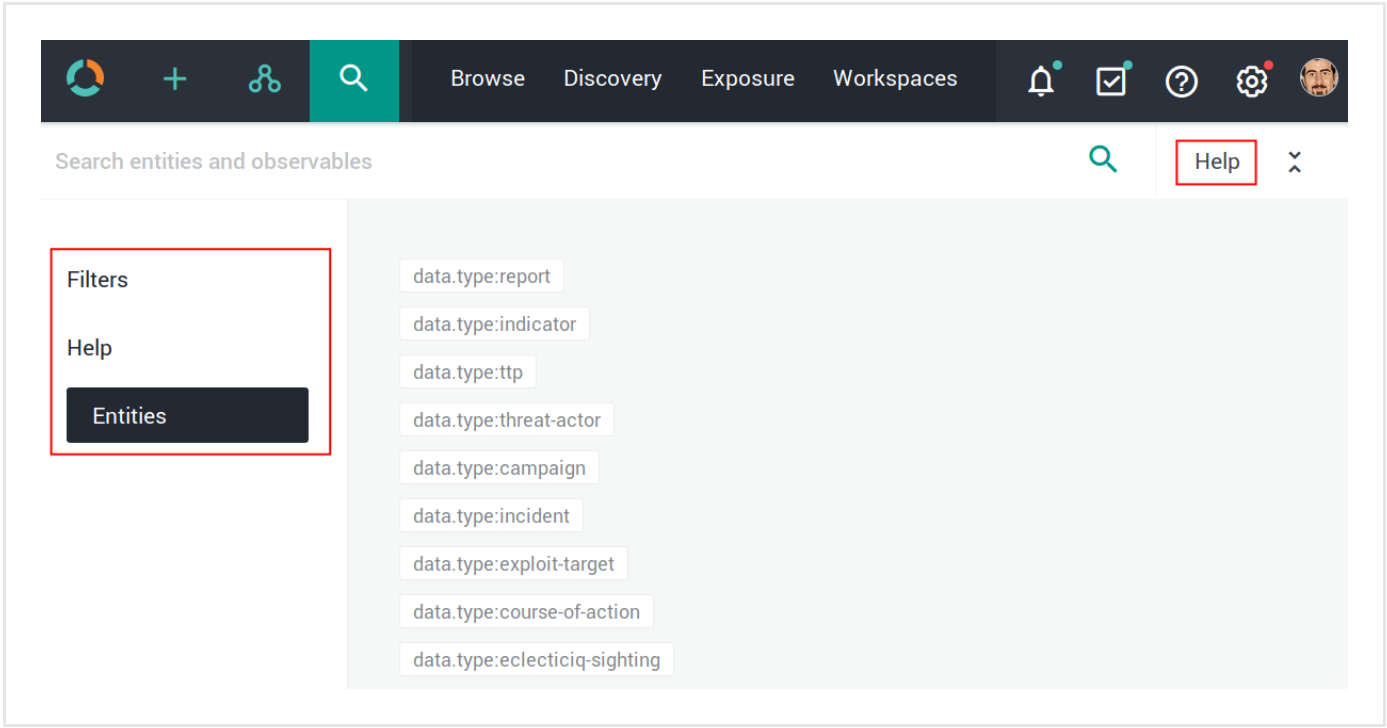


Enter search terms and search queries, and then press **ENTER** or click the search icon to run the search. Searches you run through this search box are executed platform-wide.

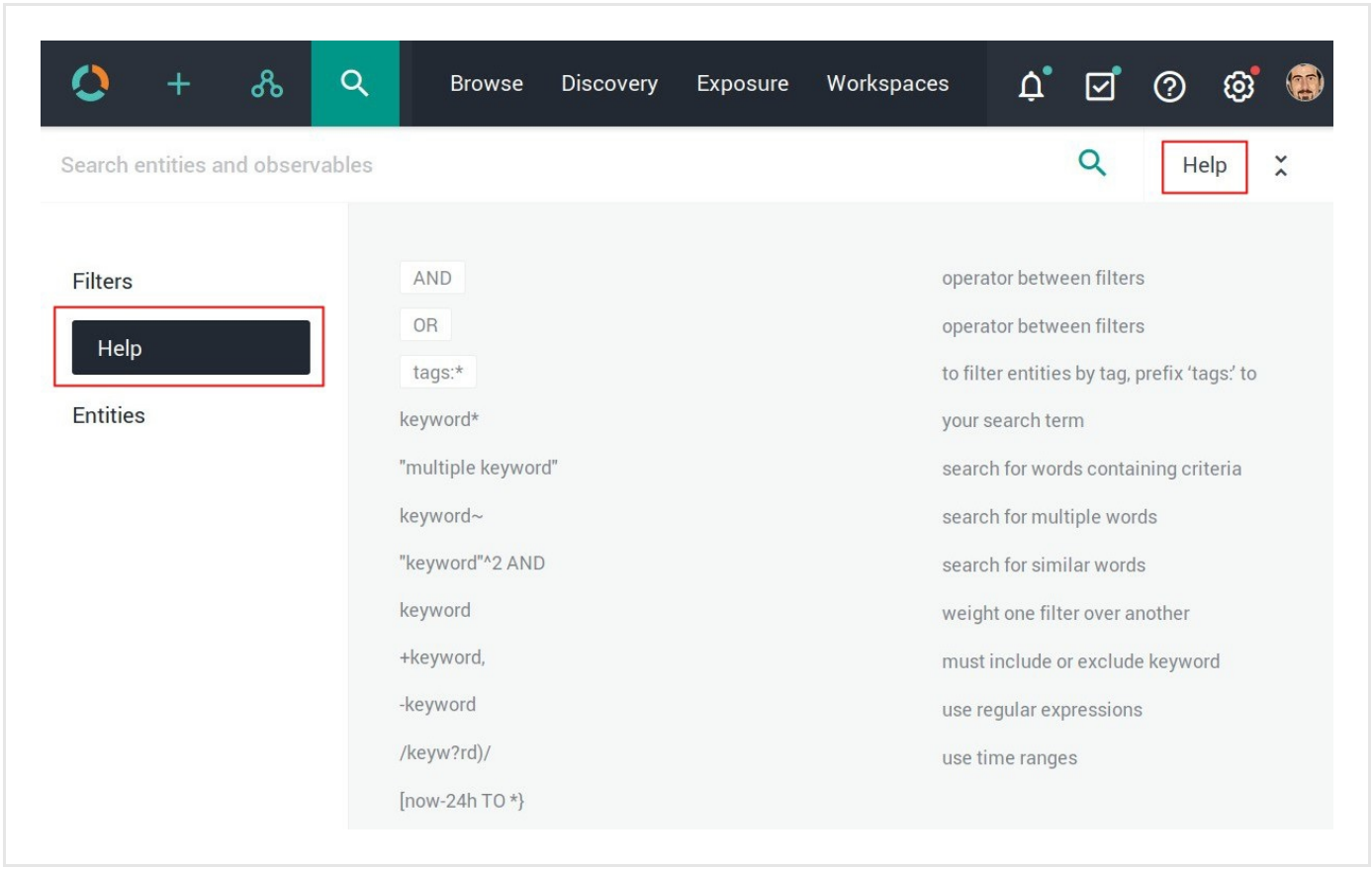
> The search functionality uses **Elasticsearch query syntax** (https://www.elastic.co/guide/en/elasticsearch/reference/current/full-text-queries.html).

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.



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



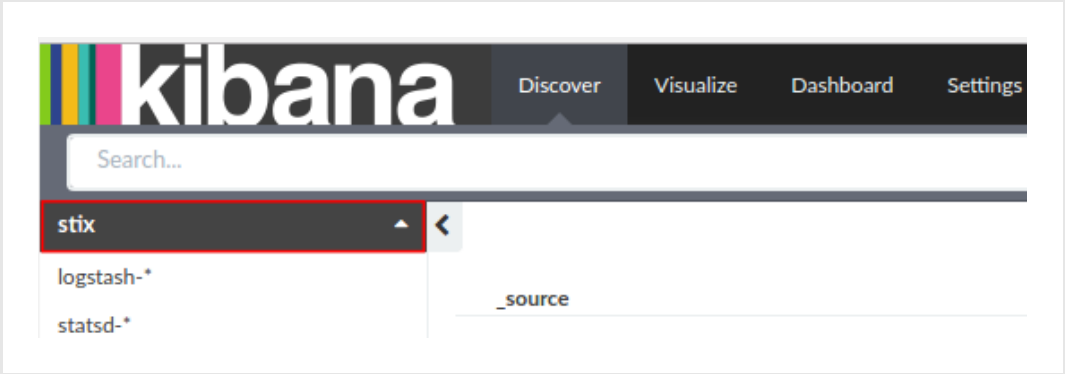
Use operators to combine multiple quick filters and create a more complex search query.
Example:

enrichment_extracts.kind:domain AND enrichment_extracts.meta.classification:high

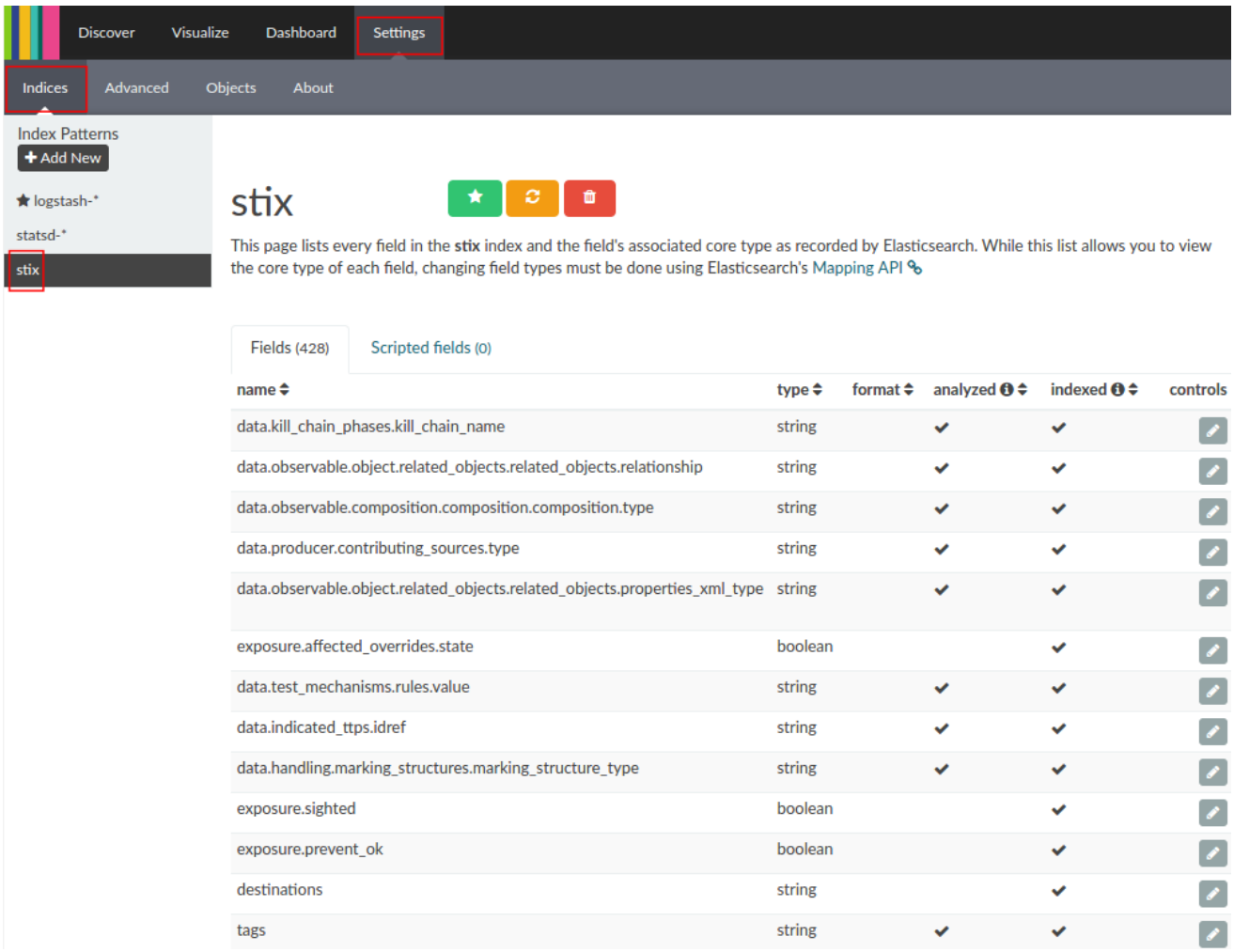
Field	Description	Example
enrichment_extracts.id	string — The alphanumeric ID string that uniquely identifies the enrichment observable.	`01h12x45-01q2-1234-od01-123456h78h90`
enrichment_extracts.kind	string — The enrichment observable data type.	`domain`
enrichment_extracts.meta.blacklisted	Boolean — An observable is blacklisted when it is included in the results returned by an *ignore* extraction rule. Allowed values: `true`, `false`.	`true`
enrichment_extracts.meta.classification	string — This value is defined in **Rules** by selecting appropriate options under **Action** and **Confidence**. Allowed classification metadata values are `good`, `bad`, and `unknown`.	`good`
enrichment_extracts.meta.confidence	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 `low`, `medium`, and `high`.	`high`
enrichment_extracts.value	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:

- Sign in to the platform with your user credentials.
- To access Kibana, in the web browser address bar enter a URL with the following format: `<platform_host>/api/kibana/app/kibana#/`.
  Keep the trailing `/`.
  Example: `https://platform.host.com/api/kibana/app/kibana#/`
- Select the **stix** index field:



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



VirusTotal integration

Integrate EclecticIQ Platform with VirusTotal to retrieve malware information about DNSs, IPs, domains, and files.

VirusTotal	integration
Integration	VirusTotal
Type	enricher/API
API endpoint	`https://www.virustotal.com/vtapi/v2/{}`
Input	ipv4, ipv6, domain, uri, hash-md5, hash-sha1, hash-sha256
Output	Enriches the 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.

EclecticIQ Platform integrates with VirusTotal to use it as an intel source. The platform integrates with the VirusTotal API through an enricher.

To enable the VirusTotal integration you configure the VirusTotal enricher as needed, and then activate it or run it manually to poll the intel source.

Configure the enricher

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 the enrichment information. Data sources can be incoming feeds, as well as other enrichers.
- Within the selected platform data sources, the entity type(s) to augment with the enrichment information.
- The enrichers to use to fetch the 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.

Observables

Observables augment the entities they are related to by providing additional context that can help discover indirect relationships or spawn new relationships between entities.
Observables are atomic and factual: an observable represents one discrete piece of information that describes a fact. For example, an IP address, a hash value, the name of a location or an actor.

Configure enricher tasks

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

- On the top navigation bar click **+ > Data management > Dataset > Enrichment** .

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Data management**.
- On the left-hand navigation sidebar click **Enrichment**.
- Click the enricher you want to configure or modify.
- On the enricher detail page, click the **Edit** button.

> ✔ On the forms, input fields marked with an asterisk are required.

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 with **Malicious - Low confidence**.
  - Enter a numeric value between *0.1* and *0.9* — that is, $0 < value < 1$.
  - Default value: *0.2*.

- **Min high confidence infection rate**: you can set a *bottom threshold* to automatically flag enriched observables with *high confidence* value.
  After completing the sample analysis, enriched observables with a *higher* detection ratio than the specified value are flagged with **Malicious - High confidence**.
  - Enter a numeric value between *0.1* and *0.9* — that is, *0 < value < 1* .
  - Default value: *0.5*.
- 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 with **Malicious - Medium confidence**.
- Click **Save** to store your changes, or **Cancel** to discard them.

Configure enricher rules

Add enricher rules

To add a new enricher rule, do the following:

- On the top navigation bar click **✚ > Rules > Enrichment**.

Alternatively:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 **+ Rule** button.

> ✔ On the forms, input fields marked with an asterisk are required.

On the **Rules > Enrichment > Create** 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 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 creation work.

Edit enricher rules

To edit enricher rules, do the following:

- On the top navigation bar, click the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 detail pane, click **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**.

> ✔ On the forms, 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 or the enricher whose observables you want to augment with additional information.
- **Entity types**: from the drop-down menu select the entity type whose observables 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 the ⚙ icon next to the user avatar image.
- From the drop-down menu select **Rules**.
- On the left-hand navigation sidebar click **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 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.
- Click **Delete** on the rule detail pane.

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.

Run the enricher

Automatically

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

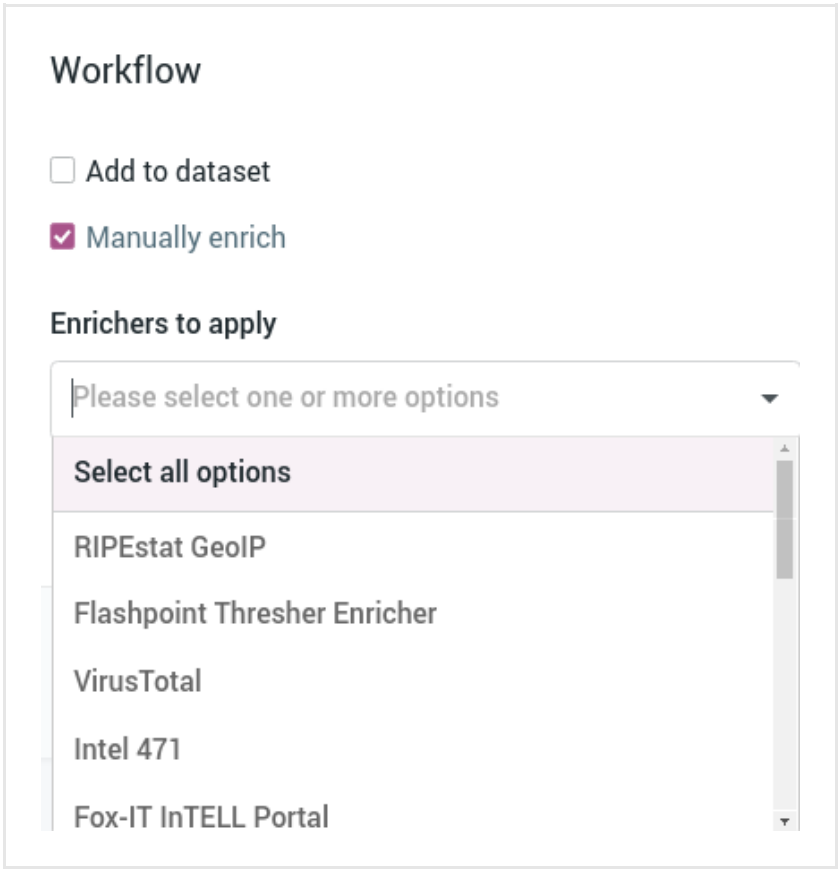
Rules give you control over the type of information you want to retrieve or exclude, and what you want to do with it. You can assign one or more enricher sources to specific observable types. You can set multiple filters to cover usage scenarios as needed. You can then examine the returned enrichment 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**.

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.

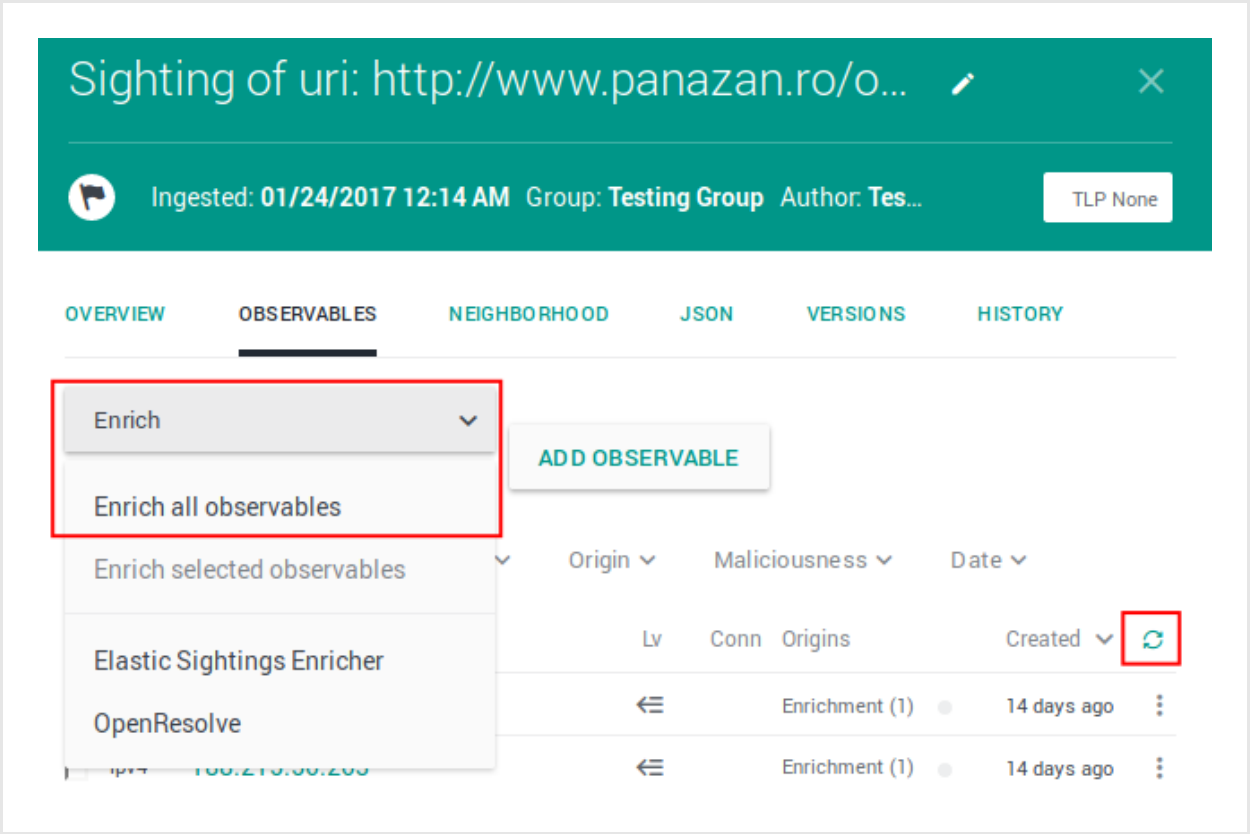
- On the entity detail pane, click **Observables**.
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.

To manually enrich the entity observables:

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

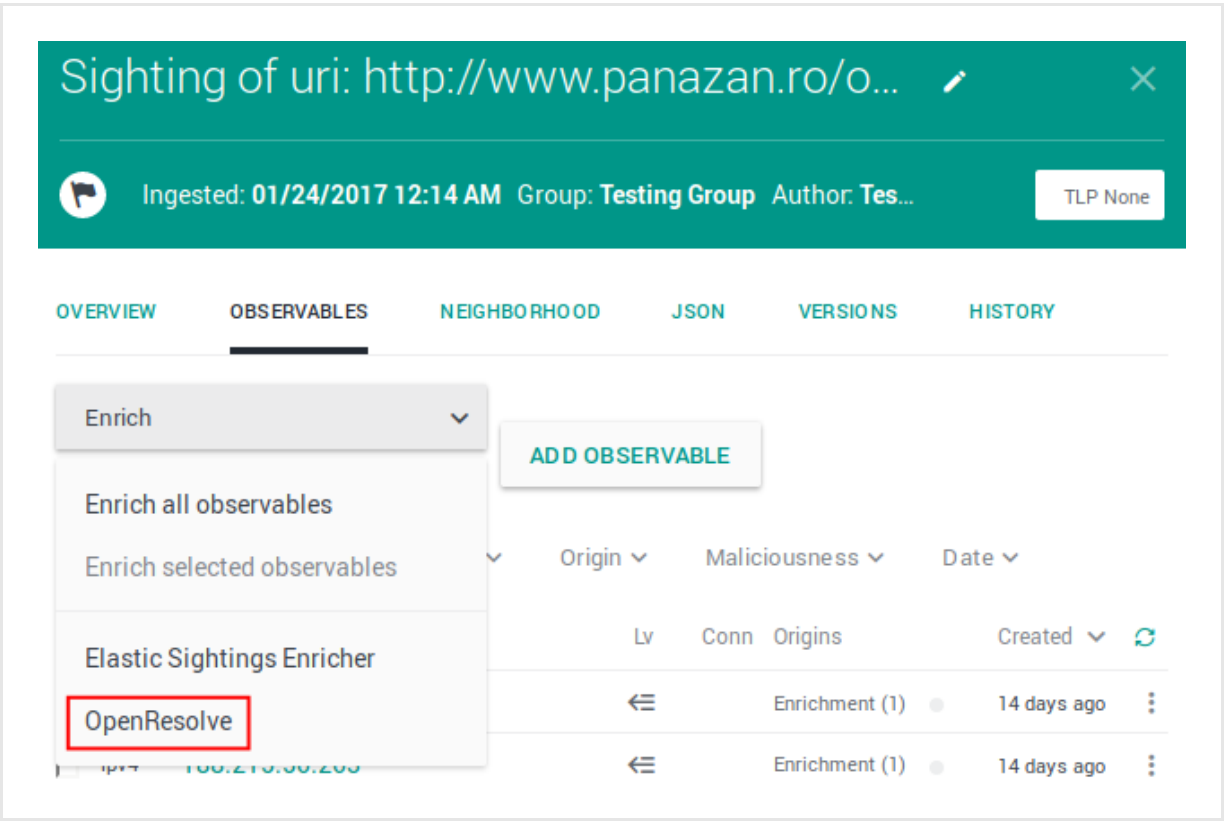
Alternatively:

- From the **Enrich** drop-down menu, select **Enrich all observables**.
- 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:

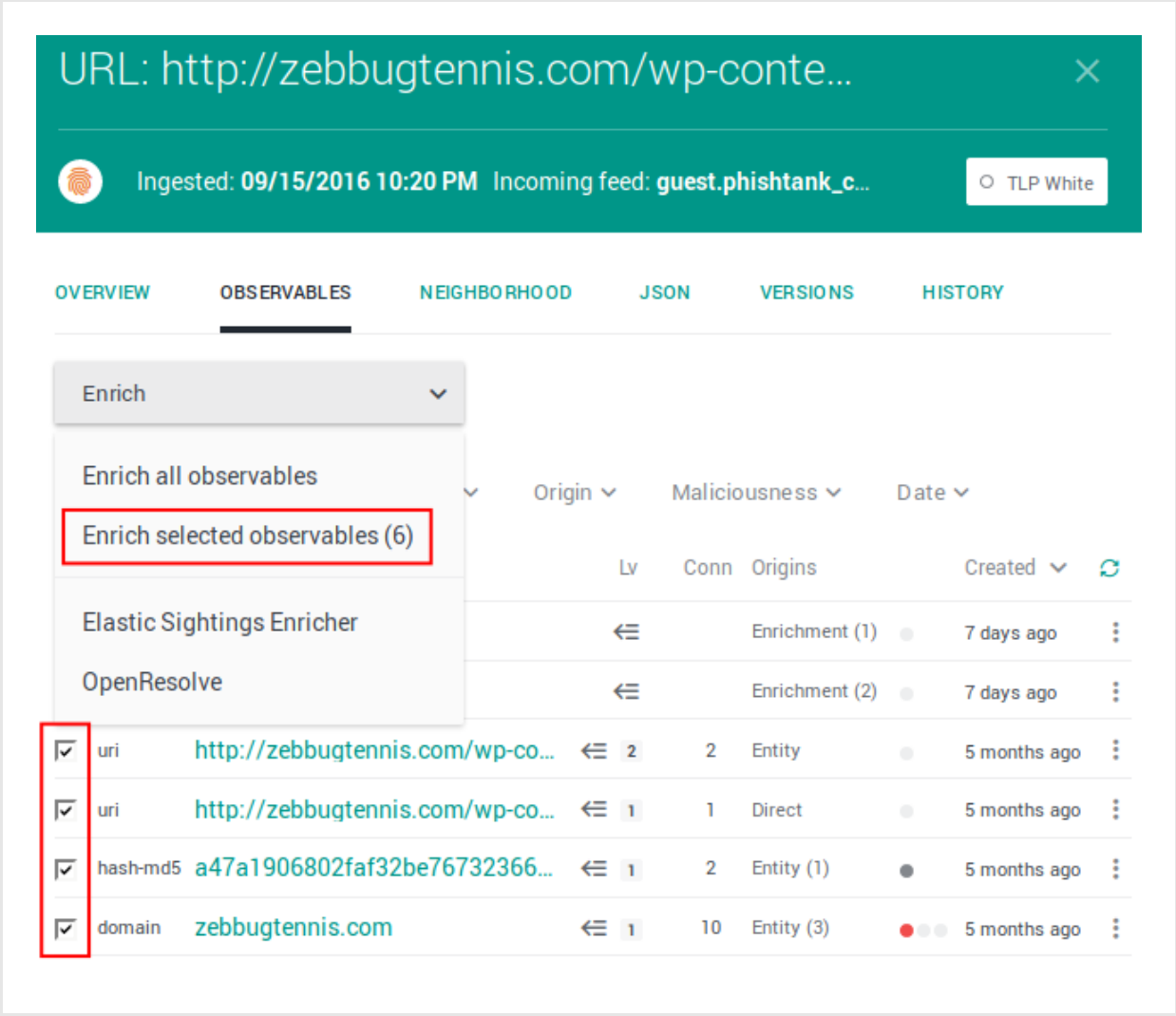
- Select it from the **Enrich** drop-down menu, and then click it.
- The platform polls the specified enricher for the entity, and it enriches all the entity observables with the retrieved data.



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 selected observables**.
- The platform polls all applicable enrichers for the entity, and it enriches the selected entity observables with the retrieved data.



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

The VirusTotal enricher can take the following observable types as input:

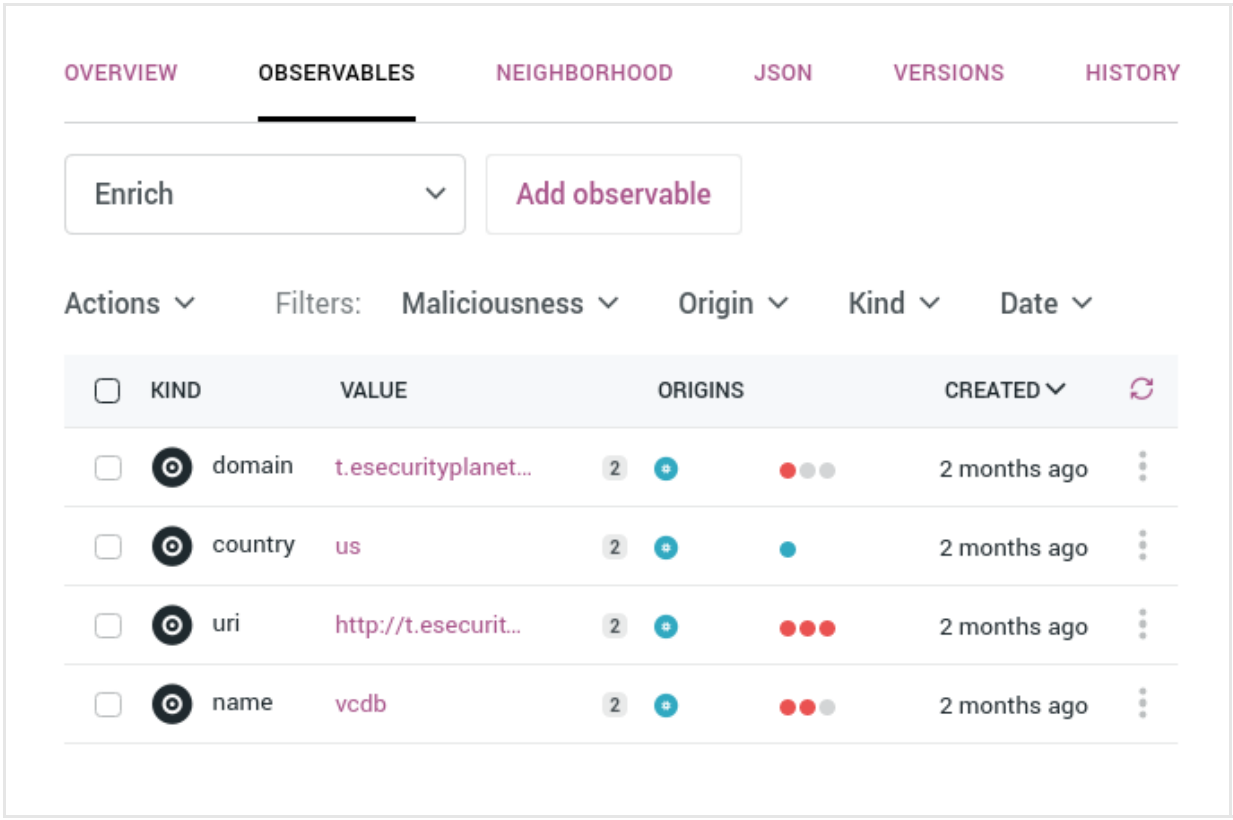
- *ipv4, ipv6, domain, uri, hash-md5, hash-sha1, hash-sha256*

The enricher uses these input data types to look for additional information to enrich existing observables with. Any entity types supporting these observable types can be enriched with VirusTotal.

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**.

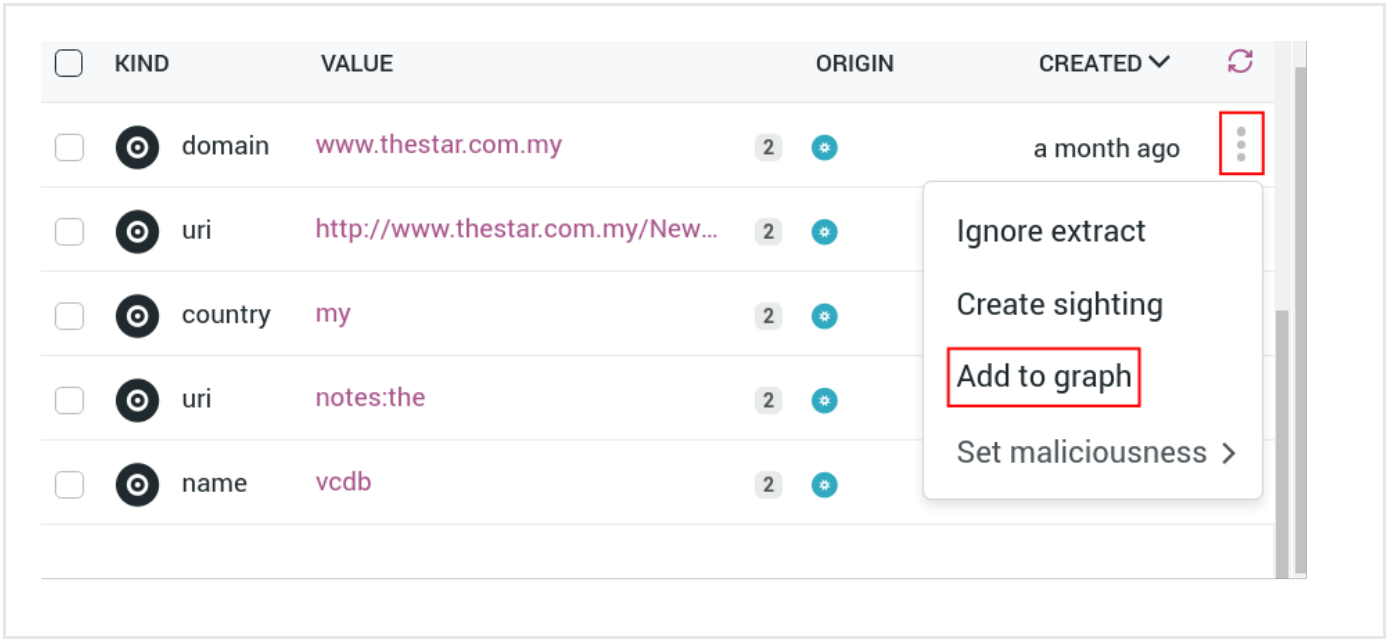
- The **Observables** tab shows an overview of the enrichment observables the entity has been augmented with.



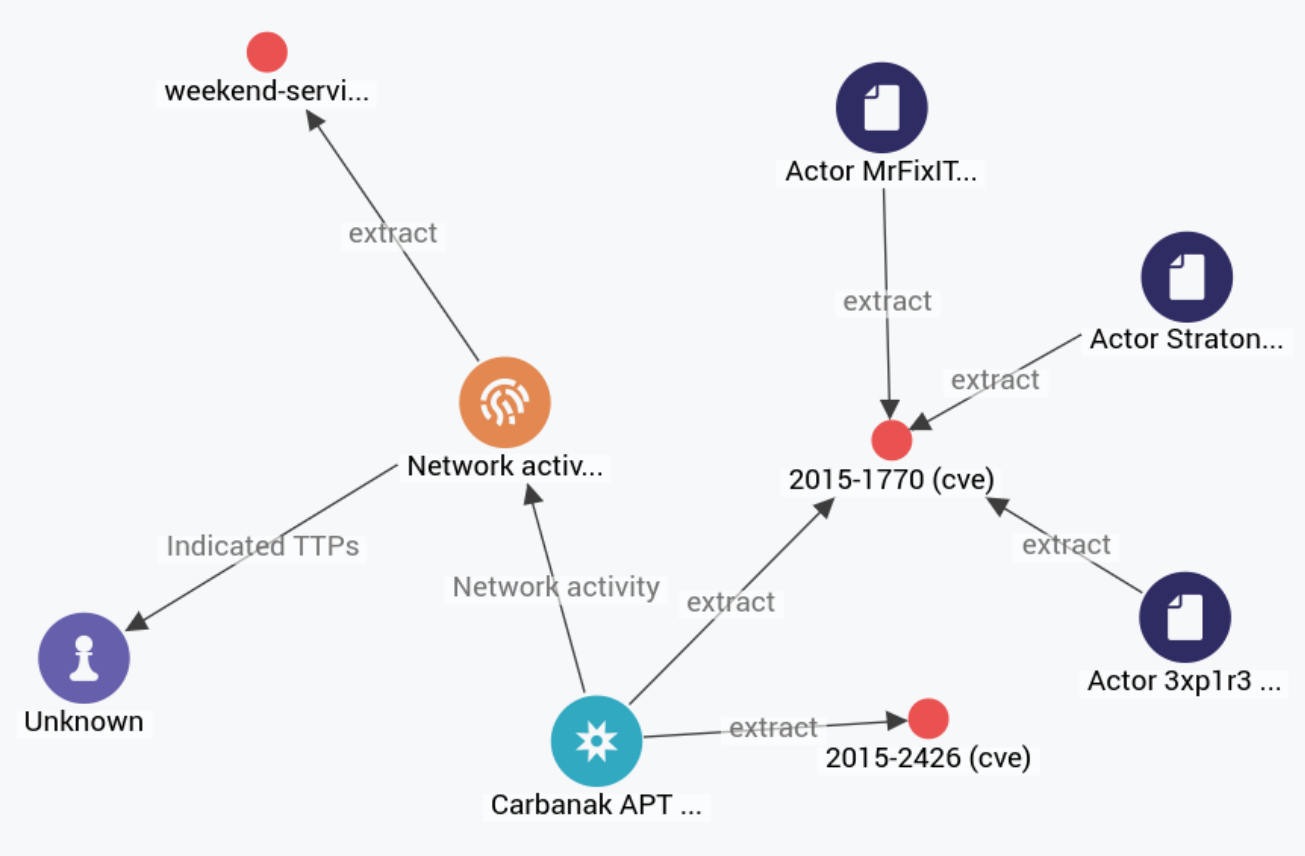
Review enrichment observables on the graph

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

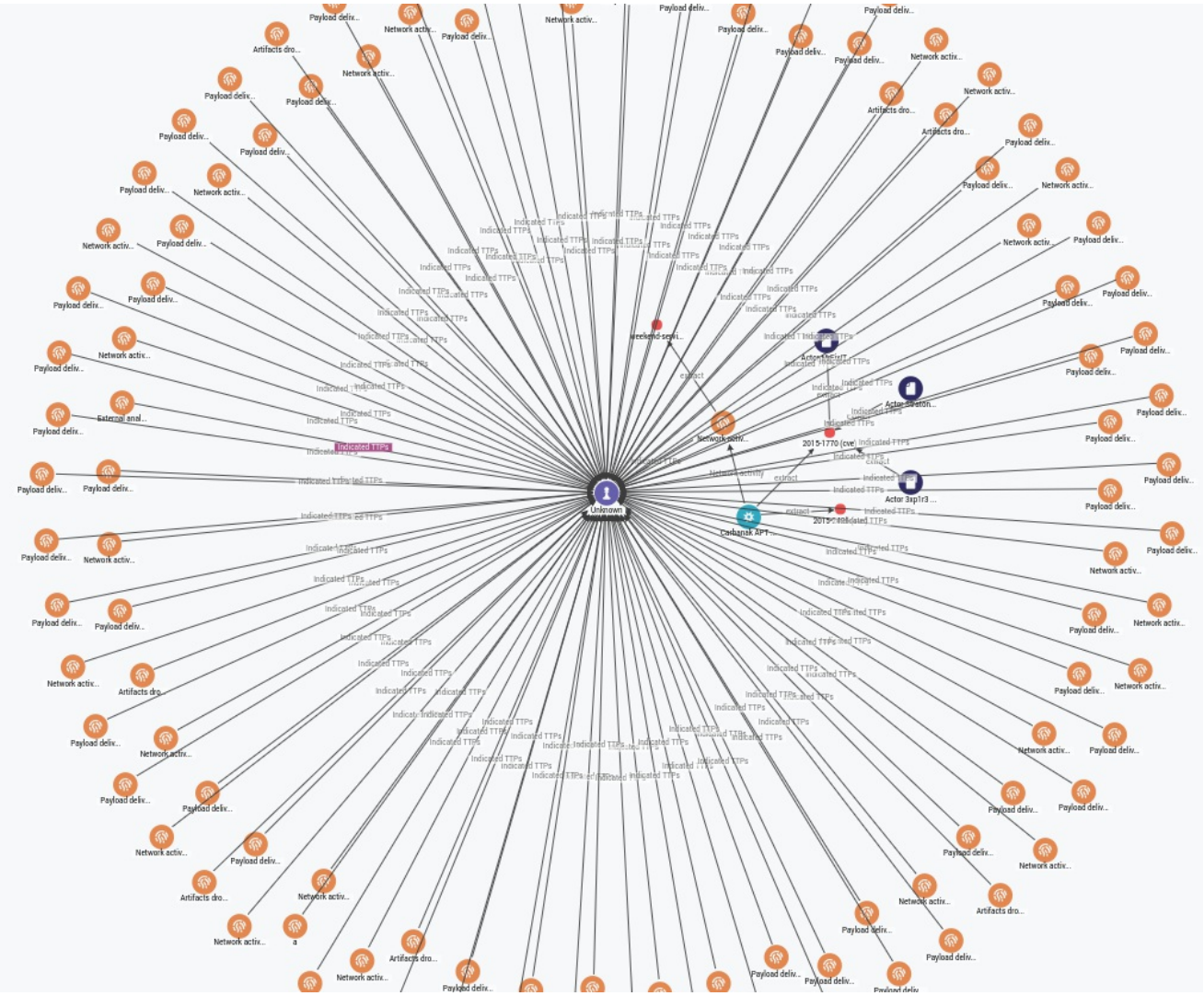
- On the row corresponding to the observable you want to load onto the graph, click the ⋮ icon, and then select **Add to graph**.



- 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** .

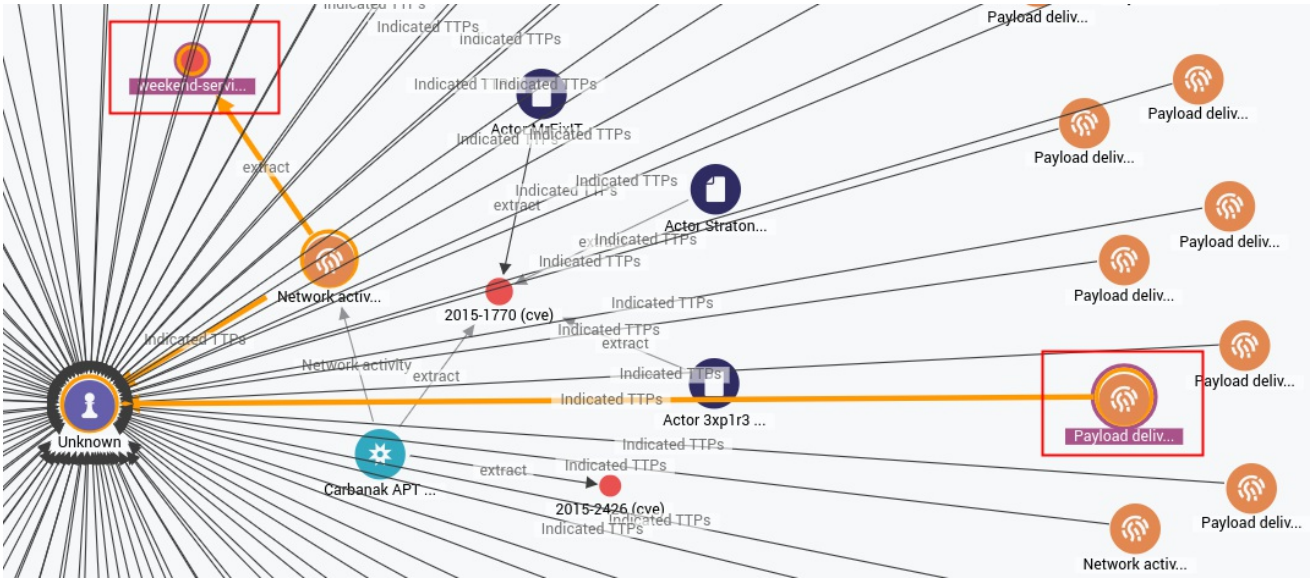


- Right-click an extract or an entity for further inspection and from the context menu select **Load entities > All**, **Load observables > All** or **Load entities by extract > All** .



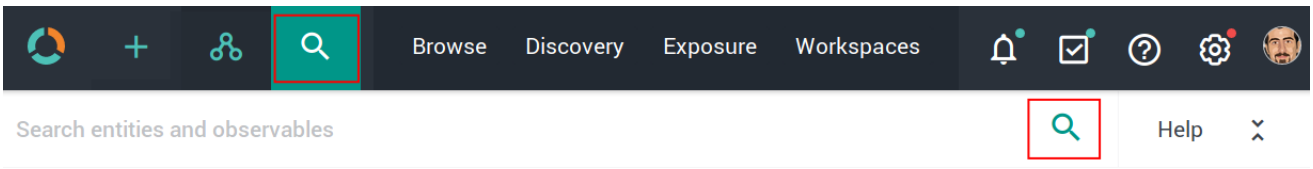
To see how entities, observables and enrichment observables are connected, and to inspect relationships between distant items, do the following:

- **CTRL** + click two nodes on the graph to select them.
- Right-click either selected node, and from the context menu select **Find path** to query the graph database about the existence of a path between the nodes, or **Show path** to highlight asn existing path on the graph.
- If a path does exist, the selected nodes and all the intermediate ones are highlighted on the graph to show the path that links them.



Search for enrichment observables

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

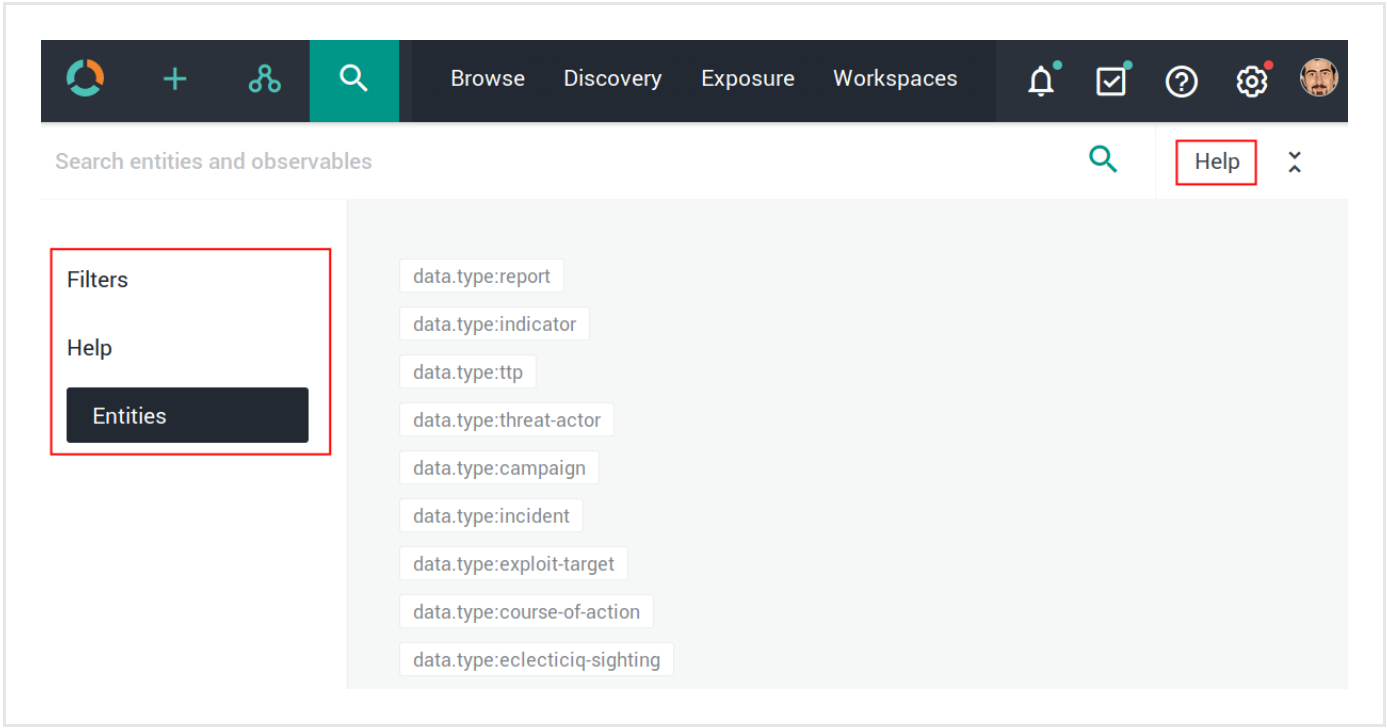


Enter search terms and search queries, and then press **ENTER** or click the search icon to run the search. Searches you run through this search box are executed platform-wide.

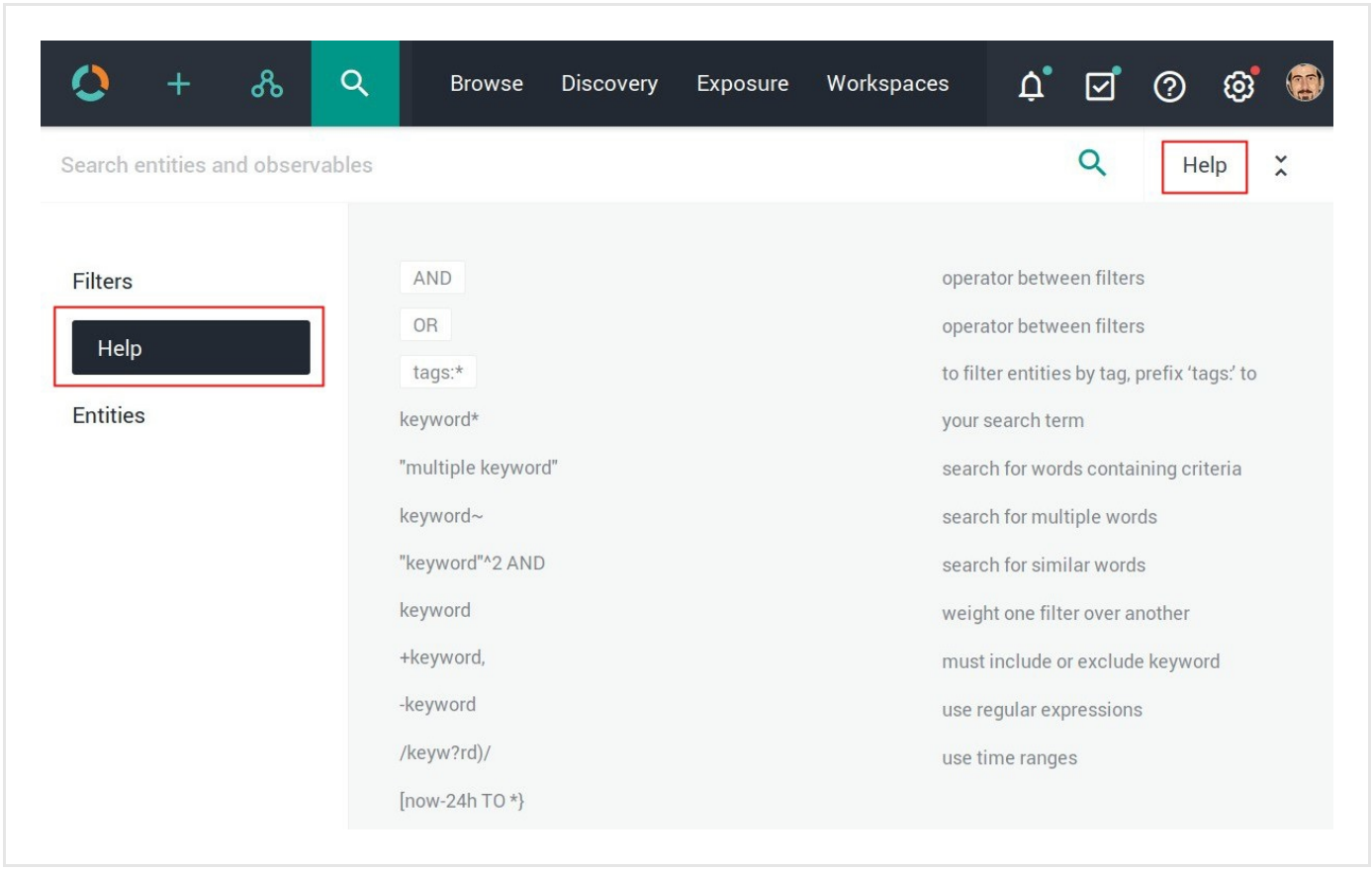
> The search functionality uses **Elasticsearch query syntax** (https://www.elastic.co/guide/en/elasticsearch/reference/current/full-text-queries.html).

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.



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



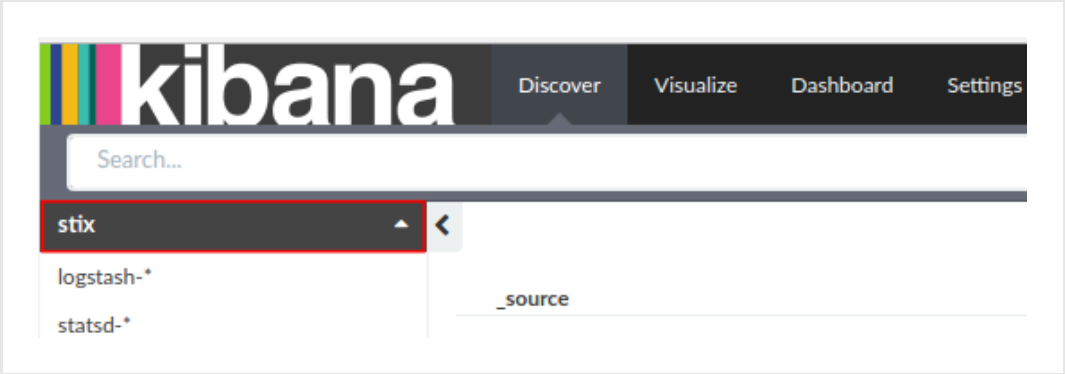
Use operators to combine multiple quick filters and create a more complex search query.
Example:

enrichment_extracts.kind:domain AND enrichment_extracts.meta.classification:high

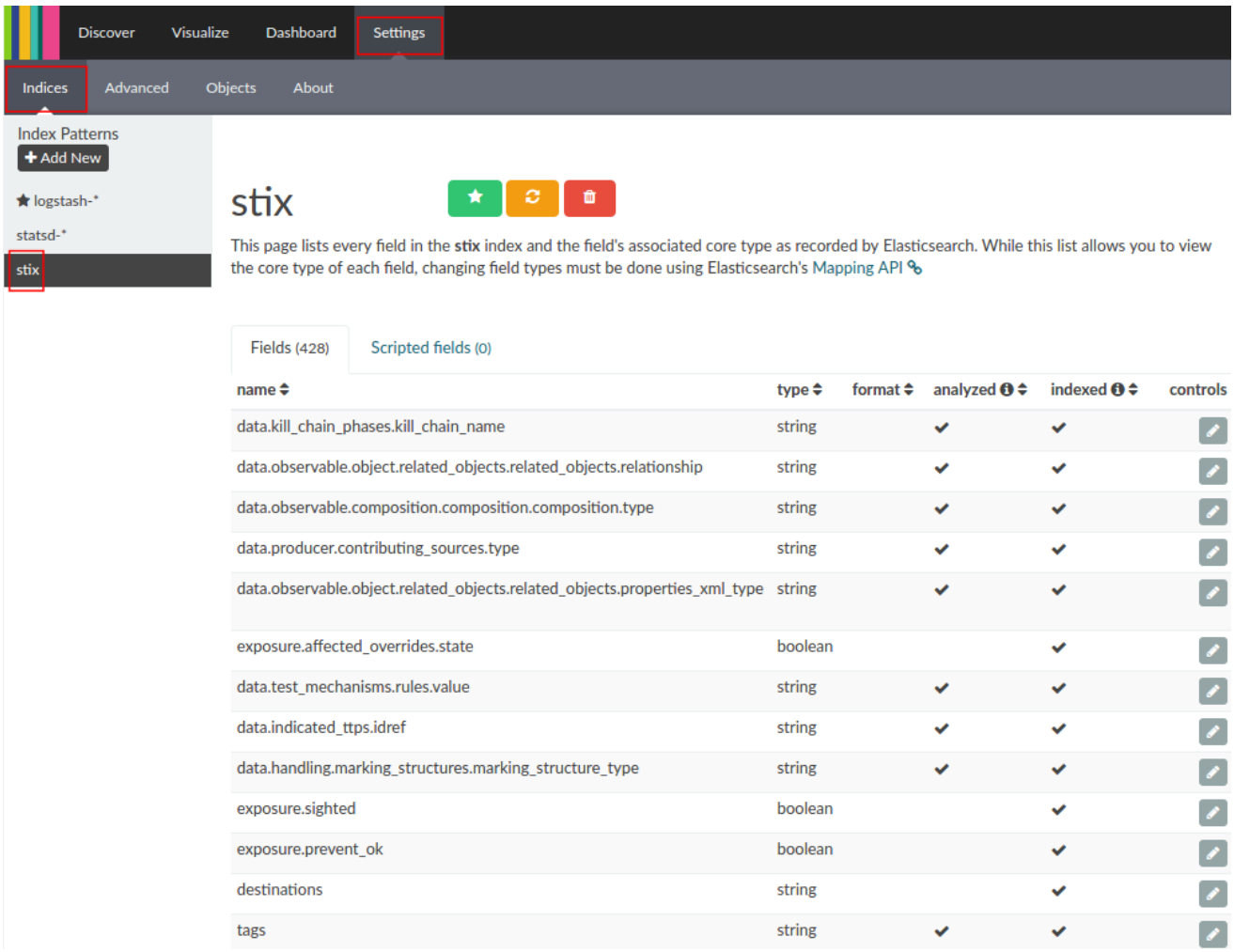
Field	Description	Example
enrichment_extracts.id	string — The alphanumeric ID string that uniquely identifies the enrichment observable.	`01h12x45-01q2-1234-od01-123456h78h90`
enrichment_extracts.kind	string — The enrichment observable data type.	`domain`
enrichment_extracts.meta.blacklisted	Boolean — An observable is blacklisted when it is included in the results returned by an *ignore* extraction rule. Allowed values: `true`, `false`.	`true`
enrichment_extracts.meta.classification	string — This value is defined in **Rules** by selecting appropriate options under **Action** and **Confidence**. Allowed classification metadata values are `good`, `bad`, and `unknown`.	`good`
enrichment_extracts.meta.confidence	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 `low`, `medium`, and `high`.	`high`
enrichment_extracts.value	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:

- Sign in to the platform with your user credentials.
- To access Kibana, in the web browser address bar enter a URL with the following format: `<platform_host>/api/kibana/app/kibana#/`.
  Keep the trailing `/`.
  Example: `https://platform.host.com/api/kibana/app/kibana#/`
- Select the **stix** index field:



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



Splunk integration

EclecticIQ Platform App for Splunk Enterprise enables Splunk users to ingest large quantities of threat intelligence by integrating EclecticIQ Platform feeds with Splunk Enterprise.

Splunk	**integration**
App	EclecticIQ Platform App for Splunk
Version	1.0.1
Compatibility	Splunk Enterprise 6.3 and later
Last changed	June 2017
Authors	SOC Prime, EclecticIQ
Type	SIEM integration
Integration	app/bidirectional
Description	The app integrates EclecticIQ Platform feeds with Splunk Enterprise. Outgoing feeds transmit relevant data to Splunk for analysis and further filtering to identify potential threats that may target your organization.
Download	**Splunkbase** `(https://splunkbase.splunk.com/app/3408/)`

Release notes

Version 1.0.1 — Several known issues were addressed.

Contact

If you want to send us your feedback or if you need any support with the app, you can contact EclecticIQ at splunk@eclecticiq.com.

To request further documentation, contact EclecticIQ at splunk@eclecticiq.com.
To suggest a feature request and to report bugs, send an email to splunk@eclecticiq.com.

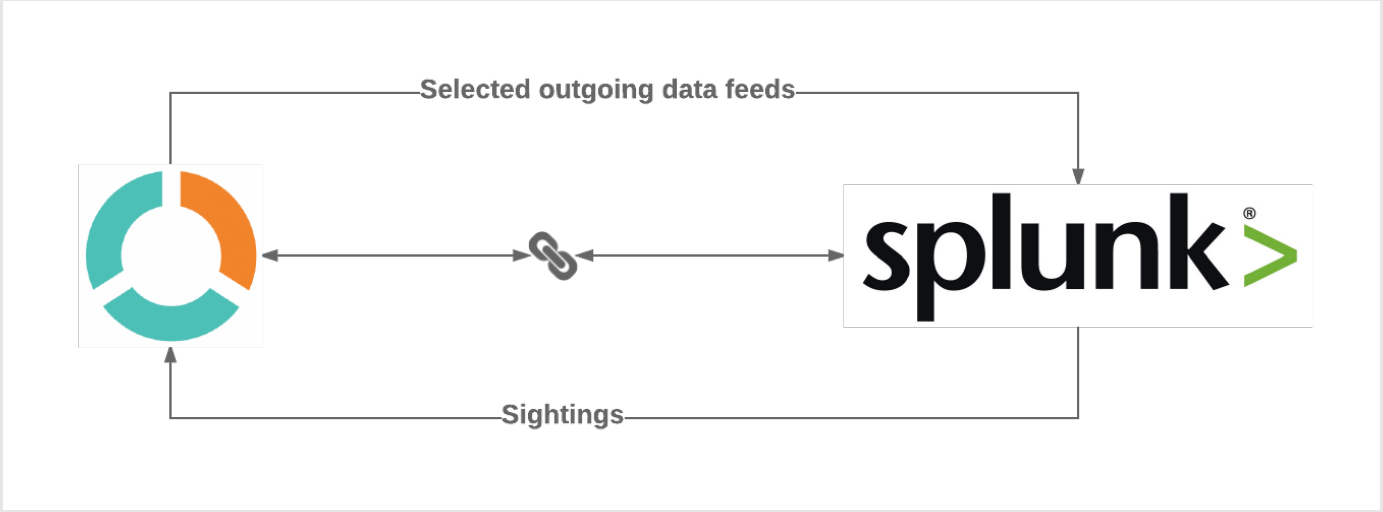
About EclecticIQ Platform App for Splunk

EclecticIQ Platform App for Splunk is an app for Splunk Enterprise. It enables Splunk users to ingest large quantities of threat intelligence by integrating EclecticIQ Platform feeds with Splunk.

EclecticIQ Platform ingests cyber threat data in different formats from multiple sources. The platform deduplicates, normalizes, and enriches source data with additional contextual details, and then it uses outgoing feeds to output relevant information to Splunk, where it can be analyzed and filtered by a set of rules to identify matching threats that may target your organization.

This process generates sightings and alerts that Splunk feeds back to EclecticIQ Platform, providing a rich threat intelligence dataset that allows you to efficiently tune your SIEM prevention and detection system.

EclecticIQ Platform App for Splunk ships with a default set of dashboard gauges to make it easier for Splunk users to monitor feed data collection, as well as to analyze and triage any *indicators of compromise* (IOCs) the data analysis process may yield.



Quick start guide

Compatibility

Splunk Enterprise 6.3 and later — EclecticIQ Platform App for Splunk 1.0.1

- EclecticIQ Platform App for Splunk 1.0.1
- Supports Splunk Enterprise 6.3 and later.
- Supports Python 2.6.6 or higher 2.x.x version.
  Not supported: Python 3.x.x.
- Required Python libraries: **argparse** `(https://pypi.python.org/pypi/argparse)`, **requests** `(https://pypi.python.org/pypi/requests)`.

Install

EclecticIQ Platform App for Splunk is developed specifically for Splunk Enterprise.
Everything you need to use the app is bundled with the installation package and the related files.
If you are using Splunk Enterprise, you do not need to install the script and configuration files.

- Verify that the Splunk Enterprise server you want to install EclecticIQ Platform App for Splunk on is compatible with the app.
- Verify that the required necessary Python libraries are installed.
- In the Splunk management console go to **Apps > Manage Apps**, and then click **Install app from file**.
- Browse to the location where the *eclecticiq-platform-app-for-splunk-<version_number>.tgz* file is stored, and then click **Upload**.
- After successfully completing the upload and the installation, restart Splunk.

Configure

After restarting Splunk, you can proceed to configuring EclecticIQ Platform App for Splunk.

- In the Splunk management console go to **Apps**.
- From the app list select **EclecticIQ Platform App for Splunk**.
- On the displayed dialog window click **Continue to app setup page**.

On the EclecticIQ Platform App for Splunk configuration screen, define the following options:

- **Feeds setup**: enter the feed ID of the EclecticIQ Platform outgoing feeds whose content you want to send to Splunk. If you enter multiple feed IDs use a comma (**,**) as a separator.
- **Input setup**: define the indexes and the source types you are using as data sources for this integration:
  - **Indexes**: enter the name of the **Splunk indexes** `(http://docs.splunk.com/splexicon:index)` you want to include as sources.
    If you enter multiple indexes, use a comma (**,**) as a separator.
  - **Sourcetypes**: enter the name of the **Splunk source types** `(https://docs.splunk.com/splexicon:sourcetype)` you want to include.
    If you enter multiple source type names, use a comma (**,**) as a separator.
- **Select the type of Sighting to send to EclecticIQ Platform**: select all applicable checkboxes corresponding to the data types you want to use to generate the sightings that are subsequently sent for ingestion to EclecticIQ Platform.
- **EclecticIQ platform URL**: enter the URL corresponding to the address of the EclecticIQ Platform host.
- **EclecticIQ source group name**: enter the name of the group you want to use as a source.
- **EclecticIQ platform authentication**: enter a valid user name and a password to authenticate and to sign in to the platform.
- Click **Save** to save and store your configuration.
- By default, a script is configured to run and collect outgoing feeds once every 2 hours at *hour:00 mins*; that is, at 00:00, 02:00, 04:00, and so on.
- By default, a script is configured to push sightings once a day at 01:00 AM.
- You can change the job schedules in the following configuration file: *$SPLUNK_HOME/etc/apps/eclecticiq-platform-app-for-splunk/default/inputs.conf*
  - *eiq_collect_feeds.py* is the script that collects outgoing feed data from EclecticIQ Platform.
  - *eiq_send_sightings.py* is the script that sends sightings to EclecticIQ Platform.

After correctly configuring EclecticIQ Platform App for Splunk to integrate and work with Splunk, the corresponding dashboard view should become populated with relevant results.

Uninstall

To uninstall EclecticIQ Platform App for Splunk, run the following command(s):

```
$ SPLUNK_HOME/bin/splunk remove app eclecticiq-platform-app-for-splunk
```

Install and configure Python

To check which Python version is installed on the target server, run the following command(s):

```
$ python -V
```

- If you need to install the required Python version, **download it** (https://www.python.org/downloads/source/), and then follow the **installation instructions** (https://docs.python.org/2/using/unix.html).
- If the required Python version is installed, check if *pip* is available on the server:

```
$ pip -V
```

- If you need to install pip, **download get-pip.py** (https://bootstrap.pypa.io/get-pip.py), and then follow the **installation instructions** (https://pip.pypa.io/en/latest/installing.html):

```
# get pip
$ wget https://bootstrap.pypa.io/get-pip.py

# install pip
$ python get-pip.py
```

- Use pip to check that the necessary libraries are available:

```
$ pip list
```

- If the *argparse* and the *requests* libraries are missing, install them:

```
$ pip install argparse
$ pip install requests
```

End of the EclecticIQ Platform App for Splunk quick start guide

---

Beginning of the EclecticIQ Platform App for Splunk integration guide

EclecticIQ Platform integration with Splunk

(Through EclecticIQ Platform App for Splunk)

Before you start

Before you start installing the app, take a moment to review the preliminary requirements and the main steps of the process.

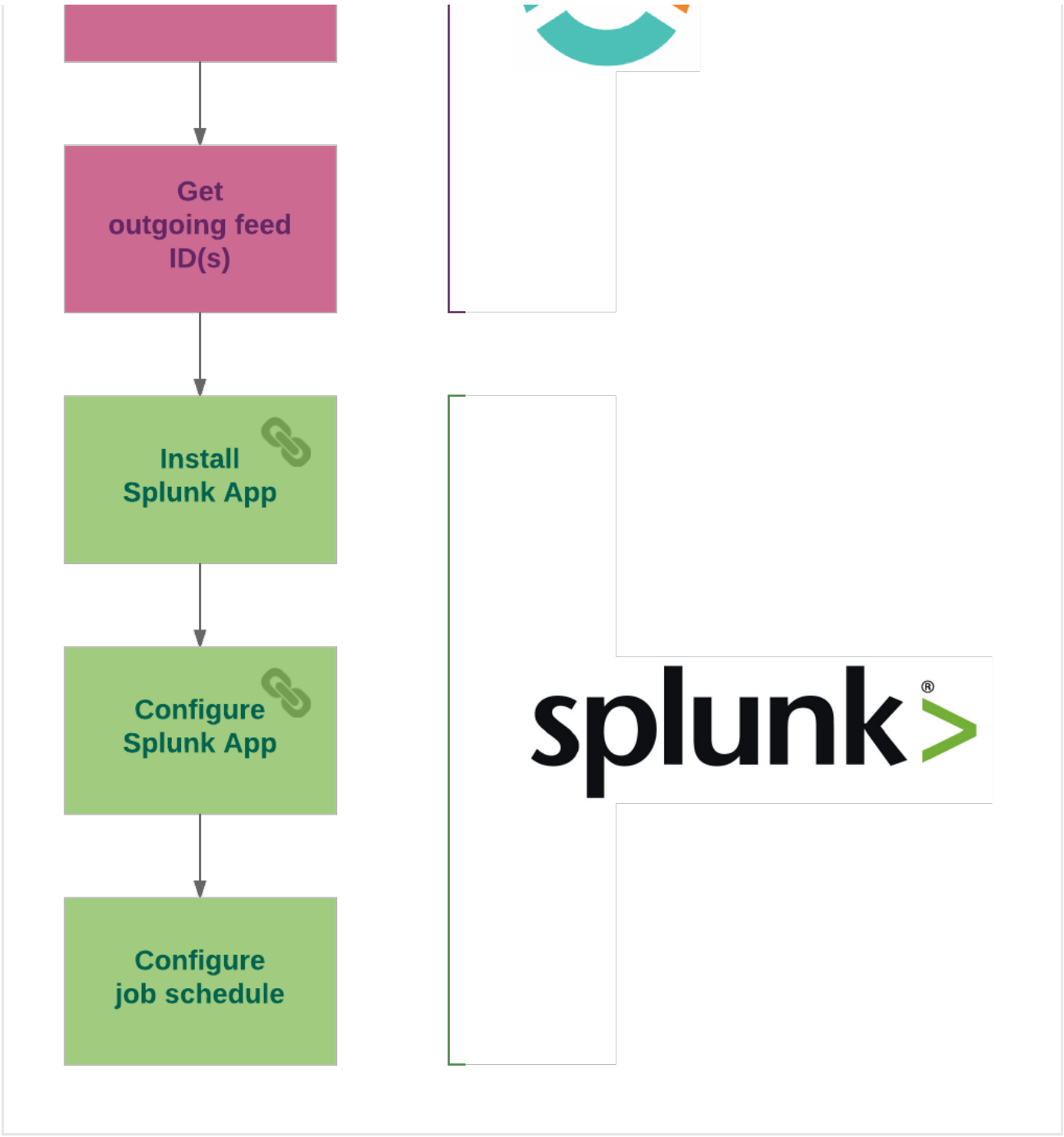
Requirements

- An EclecticIQ Platform installation.
- A Splunk server installation.
- Install and set up EclecticIQ Platform App for Splunk on a Splunk server that has network access to the EclecticIQ Platform server: these servers need to communicate and exchange data.

Process outline

The diagram sums up the main steps to set up and configure a platform integration with Splunk:

- First, you set up the outgoing feed sending data from the platform to Splunk.
- Then, you install and configure EclecticIQ Platform App for Splunk to enable the integration between the platform and Splunk.



Configure EclecticIQ CSV outgoing feeds

EclecticIQ Platform enables you to configure outgoing feeds to share and distribute cyber threat intelligence in several formats. Share knowledge and promote collaboration to support an ecosystem where partners work together to identify threats, and define an effective course of action to ensure their assets are protected.

This section describes how to configure **EclecticIQ Entities CSV** and **EclecticIQ Observables CSV** outgoing feeds, so that you can distribute selected intelligence through EclecticIQ Platform.

Configure the general options

- On the left-hand navigation sidebar click **Outgoing feeds**.
  The **Outgoing feeds** page displays an overview of the configured outgoing feeds that are made available through the platform.
- On the top-right corner of the screen, click the **+ Outgoing feed** button.

> ✔ On the forms, input fields marked with an asterisk are required.

Under **Transport and content** you can define *what* you want to publish and *how*, that is, the data content type and the data transport type.

- Under **Feed name**, enter a name for the feed you are creating. It should be descriptive and easy to remember.
- **Transport type**: from the drop-down menu select the appropriate transport type to publish data through the outgoing feed.
  You can implement this integration through the **HTTP download** and **Mount point upload** transport types.
- **Content type**: from the drop-down menu select **EclecticIQ Entities CSV** or **EclecticIQ Observables CSV** and configure the appropriate parameters under **Content configuration**, when applicable.
- **Dataset**: from the drop-down menu select one or more datasets as data sources for the outgoing feed.
- **Update strategy**: from the drop-down menu select the preferred method to update the data:
  - **Append**: every time the outgoing feed task runs, only new data from the latest task run, that is, only new entities, is appended to the existing data.
    When the outgoing feed task runs, it includes only new entities.
  - **Replace** every time the outgoing feed task runs, it publishes only new data.
    When the outgoing feed task runs, it produces new content that can include new, as well as existing entities.
  - **Diff**: every time the outgoing feed task runs, new data is compared against existing data to identify any differences between the two datasets at observable-level — any observable added to or removed from the entities in the set — or at entity-level — any entities added to or removed from the set. Depending on the selected CSV content option, each row in the CSV output contains information about one entity or one observable.
    An extra diff column is added to the output to indicate if a row, and therefore either an entity or an observable, was added to or removed from the set.
    This option allows you to identify any changes in a feed between two task runs without downloading the whole feed every time.

Set a schedule

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

- **None**: no schedule is defined. You need to manually trigger the task to ingest or to publish data through an incoming or an outgoing feed, respectively.
- **Minute**: the feed task runs automatically 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.

- **Hour**: the feed task runs automatically every hour.
  You define how long in minutes after the beginning of an hour the task should run from the corresponding drop-down menu.
- **Day**: the feed task runs automatically once a day.
  You define the time of the day when the task should run from the corresponding drop-down menu.
- **Week**: the feed task runs automatically once a week.
  You define the day of the week and time of the day when the task should run from the corresponding drop-down menu.
- **Month**: the feed task runs automatically once a month.
  You define the day of the calendar month and time of the day when the task should run from the corresponding drop-down menu.
  Keep in mind that not all months of the year have 31 days.

Set a TLP override

- **Override TLP** overwrites the **TLP** `(https://www.us-cert.gov/tlp)` color code associated to 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.
- The **Filter TLP color** options allow including in the feed data only an entity subset, based on the selected **TLP** `(https://www.us-cert.gov/tlp)` value.
  If you set a TLP color as a filter, the feed includes all the entities flagged with the selected TLP color code, as well as the entities whose TLP color indicates that they are suitable for progressively broader audiences. For example, if you select green, the feed includes entities with a TLP color set to green and entities with a TLP color set to white.

Set reliability and relevancy

- **Source reliability**: from the drop-down menu select an option to flag the content of the outgoing feed with a predefined reliability value to help other users assess how trustworthy the feed 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*
- **Relevancy threshold (%)** allows you to set a filter to include in the feed only entities whose relevancy is higher than the value defined here.

Set observable filters

- **Allowed observable states**: from the drop-down menu select one or more observable states to include in the feed data only entities whose observable states match at least one of the selections defined here.
- **Observable types**: from the drop-down menu select one or more observable types to include in the outgoing feed only entities whose observable types match at least one of the selections defined here.
- **Enrichment observable types**: from the drop-down menu select one or more enrichment observable types to include in the outgoing feed only entities whose enrichment observable types match at least one of the selections defined here.
- Click **Save** to store your changes, or **Cancel** to discard them.

The filters work independently of each other: there are no Boolean `AND` or `OR` to join multiple filters into a serial pipeline.

Save options

Besides committing 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 creation work.

Configure the transport type

Transport types	Allowed content types
HTTP download	EclecticIQ Entities CSV
	EclecticIQ Observables CSV
Mount point upload	EclecticIQ Entities CSV
	EclecticIQ Observables CSV

HTTP download

The HTTP download transport type requires basic access authentication.

If you want to make the outgoing feed data available through an HTTP URL, from the **Transport type** drop-down list select **HTTP download**.
Under **Transport configuration**, configure the following settings:

- **Public**: default setting: deselected.
  Select this checkbox to make the outgoing feed available to all platform groups and to all platform users.
  Leave it deselected to make the outgoing feed available only to specific groups. You can select the intenden recipient groups in the **Authorized groups** drop-down menu.
- **Authorized groups**: restricts access to the outgoing feed to the groups you select from the drop-down menu, and to their member users.
  The **Authorized groups** option is available only when the **Public** checkbox is deselected (default setting).

Mount point upload

If the source of the feed is located on a local or network unit, from the **Transport type** drop-down list select the **Mount point upload** option.

After selecting **Transport type > Mount point upload** , set the origin location for the source data:

- **Mount point path**: enter the path to the local or network unit where the source data for the outgoing feed is stored.

Configure the content type

When you set up an outgoing feed from the platform to the destination Splunk instance, you need to configure the following content type parameters.

From the drop-down menu select one of the following options to define the preferred structure for the output data and the resulting layout in the CSV output:

- **EclecticIQ Entities CSV**: in the resulting CSV with column headers, each row holds information referring to one entity. For example, an indicator, a TTP, and so on.
- **EclecticIQ Observables CSV**: in the resulting CSV with column headers, each row holds information referring to one observable.
  For example, an IP address, a hash, a geographic location name, and so on.

> **Warning:** If you select **EclecticIQ Observables CSV**, you need to choose at least one observable type from the **Observable types** drop-down list, and at least one enrichment observable type from the **Enrichment observable types** drop-down list.

If you select **EclecticIQ Observables CSV**, by default the outgoing feed includes only *first level*, *original* observables:

- First level: the extracted data is inside a CybOX object.
- Original: the value is extracted as is, that is, the observable holds the actual value found in the CybOX object.

You can include also *second level*, *derived* observables by selecting one or both checkboxes under **Content configuration**:

- **Include derived observables**: the extracted data is the result of an analysis of the original value found inside a STIX field.
- **Include secondary observables**: the source of the extracted data is a value inside a STIX field, not a value inside a CybOX object.

Create an automation user and group

It is a good idea to have one or more dedicated users and user groups, as necessary, to handle automation tasks that interact with external products or components of your system.
Automation groups bring together automation users, and they act as global controllers of the permissions the automation users require to operate.
Automation users handle automation and integration tasks such as data transmission through feeds and enrichers, or automatic entity creation as a follow-up action on a specific event.

Create an automation group

> **Warning:** The automation group should include all the data sources — incoming feeds, enrichers, and groups — the automation users in the group need to access.

To add a new automation user group, do the following:

- On the left-hand navigation sidebar click **System**.
- Under **User management** click **Groups**.
- Under **Groups** click the **+ Group** button.

> On the forms, input fields marked with an asterisk are required.

- Under **Create group**, define the following configuration settings:
  - **Name**: a descriptive name for the automation user group.
    Example: *DomainTools integration automation group*
  - **Description**: a short description of the automation user group and its purpose.
    Example: *Automation group for DomainTools automation users*
  - **Allowed sources**: click **+ Add** or **+ More** to add new rows as needed, where you can enter additional criteria.
    - **Sources**: from the drop-down menu select one or more data sources the automation user group and its members can access to fetch data from. The data sources can be existing incoming feeds, enrichers, as well as other user groups.
    - **TLP**: from the drop-down menu select a **Traffic Light Protocol** `(https://www.us-cert.gov/tlp)` color to filter data accordingly.
    - Click **+ Add** or **+ More** to add new rows as needed, where you can enter additional criteria.
  - **Source reliability**: from the drop-down menu select a value to filter data source reliability, so as to allow the user group to access only data from reliable sources, based on the value you set here.
  - Click **Save** to store your changes, or **Cancel** to discard them.

Save options

Besides committing 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 creation work.

Create an automation role

To add a new automation role, do the following:

- On the left-hand navigation sidebar click **System**.
- Under **User management** click **Roles**.
- Under **Roles** click the **+ Role** button.

> ✔ On the forms, input fields marked with an asterisk are required.

- Under **Create role**, define the following configuration settings:
  - **Name**: a descriptive name for the automation role.
    Example: *Rule modifier*
  - **Description**: a short description of the automation role and its purpose.
    Example: *This role allows a third-party product to modify platform rules.*
  - **Permissions**: from the drop-down menu select the actions the role is allowed to perform.

    Alternatively:
    - Start typing a permission name in the autocomplete text input field.
    - Select one or more filtered permissions from the list.
  - To revoke one or more permissions for the role, click the ✖ icon corresponding to the permission you want to remove, or the ✖ icon next to the drop-down arrow in the input field to remove all permissions at once.
  - Click **Save** to store your changes, or **Cancel** to discard them.

About permissions

- Permissions are associated to roles. Roles act as containers for sets of permissions defining the scope of actions of the corresponding roles.
- Permissions are predefined in the platform, and they are not editable or configurable. You can either grant them to roles, or revoke them.
- Permission names strive to be self-explanatory:
  Format: *<type of action> <object of the action>*
  Example: *modify entities*
- Permissions allow two types of action:
  - **modify**: a modification permission that allows write operations.
  - **read**: a read permission that grants access to data without allowing any modifications.

To get an overview of the available permissions available on the platform, do the following:

- On the left-hand navigation sidebar click **System**.

- Under **User management > Permissions**, the permission overview is displayed as a table, where each permission is assigned a row.

Create an automation user

To add an automation user, do the following:

- On the left-hand navigation sidebar click **System**.
- Under **User management** click the **User** button.

> ✔ On the forms, input fields marked with an asterisk are required.

In the user editor define the following configuration settings:

- **First name**: enter a name that provides a short description of the automation role and its purpose.
- **Last name**: enter a name that provides a short description of the automation role and its purpose.
- **User name**: enter the designated user name to identify the user, when signed in to the platform.
  Choose a name that helps understand what the automation user does.
  Example: *DomainTools integration connector*
- **Email**: an email address associated to the automation user. You can use this address to send and receive automated notifications.
- **Active**: select this checkbox to enable the user immediately after saving the newly created user profile.
  Active users can sign in to the platform and carry out actions, based on their user profile and their permissions.
- **Administrator**: select this checkbox to elevate the user's role to administrator.
  When the checkbox is selected, the user has full administrator rights and permissions.
- **Contact info**: n/a
- **PGP public key**: the user's **PGP public key** `(https://ssd.eff.org/en/module/introduction-public-key-cryptography-and-pgp)`.
- **Locale**: from the drop-down menu select the appropriate **locale** `(https://en.wikipedia.org/wiki/locale_(computer_software))` settings for the user interface.
- **Use system timezone**: select this checkbox to override any locale-specific time zone setting with the system-defined time zone.
- **Groups**: from the drop-down menu select one or more groups to assign the new user to.
  Alternatively, search for a group by starting typing a group name in the autocomplete text input field.
- To remove the user from one or more groups, remove the relevant entries by clicking the ✖ corresponding to the group you want to remove the user from.
- **Roles**: it works like **Groups**, the only difference being that instead of adding the user to one or more groups, this option assigns one or more roles to the user.
- Click **Save** to store your changes, or **Cancel** to discard them.

Get the automation group meta.source ID

Platform entities include a `meta.source` property key/value pair to identify the platform group as a data source.
If you want to programmatically create entities in the platform, you need to pass a group `meta.source` ID value when you make the corresponding calls to the platform API.
Likewise, if you want to identify the platform source group an entity comes from when the platform transmits data to an external product or system, you can retrieve the `meta.source` property key/value pair.

To retrieve the correct `meta.source` ID value related to an automation group, do the following:

- Get the automation group ID.
- Pass the automation group ID to get the `meta.source` ID.

Step 1 of 2: get the group ID

To retrieve the automation group ID value you need, so that you can retrieve the `meta.source` ID you pass in the calls to the platform API, do the following:

- On the left-hand navigation sidebar click **System**.
- Under **User management** click **Groups**.
- On the platform group overview page, click the row corresponding to the automation group associated to the data source(s) you want to use as input *and* to the automation user making the API calls.
- The action returns a URL with the following format: *https://<platform_host>/user-management/groups?detail=<integer>*
  Example: *https://platform.host/user-management/groups?detail=30*

In the example, the `detail` value is `30`. This is the group ID.
You need to pass this value in a call to a specific platform API endpoint to retrieve the `meta.source` ID.

Step 2 of 2: get the group source ID

To retrieve the `meta.source` ID to make calls to the platform API to programmatically create entities, do the following:

- Make an authentication call to to the platform API to validate your user credentials and to receive a Bearer token.
- Make a call to the `/api/groups/<group_ID>` endpoint:
  - Include the Bearer token in a `Bearer` header in the call.
  - Include the group ID you previously retrieved as a trailing element in the URL.
    Example: `https://platform.host/api/groups/30`
- In the JSON response, look for the group object with the `"id" : <integer>` key/value pair matching the group ID you previously retrieved.
  Example: `"id" : 30,`
- In the same group object, look for the `"source" : "<UUID_string>"` key/value pair.
  This is the group `meta.source` ID you need to pass in API calls to programmatically create entities.

Get the automation group meta.source ID example

Get the group ID

- On the platform group overview page, click the row corresponding to the automation group associated to the data source(s) you want to use as input.

```
https://platform.example.com/user-management/groups?detail=30
```

cURL API request — fetches the group meta.source

```
$ curl -X GET
       -v
       --insecure
       -i
       -H "Content-Type: application/json"
       -H "Accept: application/json"
       -H "Authorization: Bearer <token>"
       https://platform.host/api/groups/30

# copy-paste version:
$ curl -X GET -v --insecure -i -H "Content-Type: application/json" -H "Accept: application/json"
-H "Authorization: Bearer <token>" https://platform.host/api/groups/30
```

API response — returns the group meta.source

```
{
  // Number of returned user groups
  "count": 18,

  "data": [

    ...

    {
      "allowed_sources": [

        // Lists all allowed data sources configured in the group editor
        ...

      ],

      // Group id, same value as the 'detail=' URL param for the group
      "id": 30,

      // Group 'meta.source' ID you need to pass in API calls
      "source": "42c051f8-9f5b-4696-a629-b86c2ead955f",

      // Group 'meta.source_name', the group name defined in the group editor
      "name": "DomainTools automation group",

      "type": "groups",
      "users": [

      // Lists all users that are part of the group
      ...

      ]
    },

    ...

  ]
}
```

Authentication

The authentication mechanism is based on **JSON web tokens** `(http://jwt.io/)`.

By default, the token expires 30 minutes after successfully signing in. The corresponding session is terminated, and you need to sign back in to the platform.
When human interaction is detected — for example, keystrokes or mouse activity — the token is automatically refreshed every 2 minutes. This prevents the system from signing out users who may be working or saving data at that time.
Therefore, the default maximum amount of idle time without any human interaction before being automatically signed out equals to *session token validity - 2 minutes*.

To authenticate and access the platform, do the following:

- Make a `POST` call.
- In the call, pass your authentication credentials as a JSON object to the `/auth` endpoint. The credential data is used to generate a token that is returned with the response.

You need to include the generated bearer token in the `Authorization` HTTP header with each subsequent API call.
The `Authorization` HTTP header has the following format: `Authorization: Bearer <token>`

Auth request

API endpoint	`/auth`
Auth method	`POST`
HTTP headers	`"Content-Type: application/json", "Accept: application/json"`
API request	`POST + "Content-Type: application/json" + "Accept: application/json" + { "username": "<valid_user_name>", "password": "<valid_password>" } + <platform_host>/auth`
API response	`{ "expires_at": "<expiry timestamp>", "token": "<token>" }`

The following example uses cURL to authenticate:

```
$ curl -X POST
       --insecure
       -H "Content-Type: application/json"
       -d '{ "username" : "<valid_user_name>", "password" : "<valid_password>" }'
       https://platform.host/api/auth

# copy-paste version:
$ curl -X POST --insecure -H "Content-Type: application/json" -d '{ "username" : "
<valid_user_name>", "password" : "<valid_password>" }' https://platform.host/api/auth
```

Auth response

When the user name and password credential are valid, the `POST` call returns a JSON web token:

```
{
  "expires_at": "2016-03-30T12:11:40.078219+00:00",
  "token"     :
"abHpYXQiOjE0NTkzMzI3MDAsIm4TcCI6MTQ1OTMzOTkwMCwiYWxnIjoiSFMyNTYifQ.oyY1c2VyX2lkIjo1fQ.LQQ3NdUHp4s-
QCXsxq3feI0Dy6tf5XQX9DOML1RNIzQ"
}
```

You need to include the bearer token value in each subsequent API call. You pass the token by including an `Authorization` HTTP header in the API request.
The `Authorization` HTTP header has the following format: `Authorization: Bearer <token>`

In the following example, you make a `GET` request to the `/api/` endpoint to retrieve a list of the available API endpoints and the corresponding methods:

```
$ curl -X GET
       -v
       --insecure
       -i
       -H "Content-Type: application/json"
       -H "Accept: application/json"
       -H "Authorization: Bearer <token>"
       https://platform.host/api/

# copy-paste version:
$ curl -X GET -v --insecure -i -H "Content-Type: application/json" -H "Accept: application/json"
-H "Authorization: Bearer <token>" https://platform.host/api/
```

Warning:
About cURL calls

- If you make HTTPs cURL calls to the API *and* you have a self-signed or an invalid certificate, include the `-k` or the `--insecure` parameter in the cURL call to skip the SSL connection CA certificate check.
- Always append a `/` trailing slash at the end of an API URL endpoint. The only exception is `/auth`, which does not take a trailing slash.
- In the cURL call, the `-d` data payload with the entity information always needs to be flat JSON, not hierarchical JSON.
  If you want to pass a hierarchical JSON object, include the `--data-binary` parameter, followed by the path to the JSON file, for example `@/path/to/entity_file.json`.

Get the feed ID

You can access and download content from an outgoing feed by specifying its ID.
A feed ID is included in the outgoing feed URL as a URL parameter.

Get the feed ID through the GUI

To get the feed ID through the platform GUI, do the following:

- On the left-hand navigation sidebar click **Outgoing feeds**.
- On the **Outgoing feeds** overview, browse to the feed whose ID you need to retrieve, and then click the corresponding row.
- The outgoing feed URL is loaded on the web browser address bar. For example:
  `https://<platform.domain>/#/configuration/outgoing-feeds?detail=78&tab=detail`
- The `detail` URL parameter holds the feed ID.
  In the example URL, `detail=78` indicates that the selected outgoing feed ID is `78`.
  When you make an API call to retrieve the feed content, you need to include the ID value in the API endpoint.

Get the feed ID through the API

Make an API call to download a list of all available public outgoing feeds.
This call returns a JSON object with an array listing all available public outgoing feeds with HTTP transport type.

API endpoint	`/open-outgoing-feed-download/`
API method	`GET`
HTTP headers	`"Content-Type: application/json"`, `"Accept: application/json"`, `"Authorization: Bearer <token>"`
API request	`GET` + `"Content-Type: application/json"` + `"Accept: application/json"` + `"Authorization: Bearer <token>"` + `<platform_host>/open-outgoing-feed-download/`
API response	`{ "data" : [ <open_outgoing_feed_array> ] }`

API request outgoing feeds

cURL call

```
$ curl -X GET
       -v
       --insecure
       -i
       -H "Content-Type: application/json"
       -H "Accept: application/json"
       -H "Authorization: Bearer <token>"
       https://platform.host/api/open-outgoing-feed-download/

# copy-paste version:
$ curl -X GET -v --insecure -i -H "Content-Type: application/json" -H "Accept: application/json"
-H "Authorization: Bearer <token>"
```

API response outgoing feeds

```
{
  "data": [

    {
      "id": 1,
      "link": "/api/open-outgoing-feed-download/1",
      "name": "Default outgoing feed"
    },

    {
      "id": 16,
      "link": "/api/open-outgoing-feed-download/18",
      "name": "Public feed with electrolytes"
    },

    {
      "id": 25,
      "link": "/api/open-outgoing-feed-download/25",
      "name": "XYZ"
    }

  ]
}
```


Get a specific outgoing feed

Make an API call to download the details of a specific outgoing feed.
This call returns a JSON object containing the details of a specific public outgoing feed with HTTP transport type.
To select the public outgoing feed whose details you want to retrieve, include the feed ID in the API request endpoint.

API endpoint	`/open-outgoing-feed-download/<feed-id>/`
API method	`GET`
HTTP headers	`"Content-Type: application/json"`, `"Accept: application/json"`, `"Authorization: Bearer <token>"`
API request	`GET` + `"Content-Type: application/json"` + `"Accept: application/json"` + `"Authorization: Bearer <token>"` + `<platform_host>/open-outgoing-feed-download/<feed-id>/`
API response	`{ "data" : { <specific_feed_details> } }`

API request specific outgoing feed

cURL call

```
$ curl -X GET
       -v
       --insecure
       -i
       -H "Content-Type: application/json"
       -H "Accept: application/json"
       -H "Authorization: Bearer <token>"
       https://platform.host/api/open-outgoing-feed-download/18

# copy-paste version:
$ curl -X GET -v --insecure -i -H "Content-Type: application/json" -H "Accept: application/json"
-H "Authorization: Bearer <token>" https://platform.host/api/open-outgoing-feed-download/18
```

API response specific outgoing feed

The response details include an array listing the successful feed executions.

The paths in the `content_blocks` array have the following format:
`/api/open-outgoing-feed-download/<feed-id>/runs/<run-id>/content-blocks/<content-block-id>`

- A *run* is a feed execution to publish the feed content.
- A *content block* is a data blob whose format depends on the content type defined for the feed. For example, JSON, CSV or STIX.


```
{
  "data": {

    "content_blocks": [
      "/api/open-outgoing-feed-download/18/runs/0ad2edd4-8a7b-4894-b8b3-aee90a22ebaa/content-
blocks/32",
      "/api/open-outgoing-feed-download/18/runs/5fdeff71-93af-43a5-b94e-c4ab857a749c/content-
blocks/33",
      "/api/open-outgoing-feed-download/18/runs/40e31ada-06e6-4647-a287-4c9b54841619/content-
blocks/34",
      "/api/open-outgoing-feed-download/18/runs/0f56ec9c-cc1e-4aae-afd0-f693f412ad55/content-
blocks/35",
      "/api/open-outgoing-feed-download/18/runs/d842dd68-8ecf-4ecf-b073-a591d361cf26/content-
blocks/36",
      "/api/open-outgoing-feed-download/18/runs/eed28e1e-4352-42a5-8b1f-cfc918b0e0ab/content-
blocks/37",
      "/api/open-outgoing-feed-download/18/runs/f830aa7b-4ddc-4725-b13c-7cbe445f306d/content-
blocks/40",
      "/api/open-outgoing-feed-download/18/runs/a11bb585-720a-4c56-b650-90cb9d6a69e5/content-
blocks/41",
      "/api/open-outgoing-feed-download/18/runs/6e677f4b-c91d-49dd-9c39-70266987b863/content-
blocks/42"
    ],

    "id": 18,
    "name": "Public feed with electrolytes"
  }
}
```


Install and configure EclecticIQ Platform App for Splunk

EclecticIQ Platform App for Splunk is a native application that installs directly on your Splunk instance.
This section describes how to download and install EclecticIQ Platform App for Splunk, as well as how to configure Splunk to work with the app.

Download the app

- Download the *eclecticiq-platform-app-for-splunk-<version_number>.tgz* file from **Splunkbase** `(https://splunkbase.splunk.com/app/3408/)`.
- Save the archive locally.

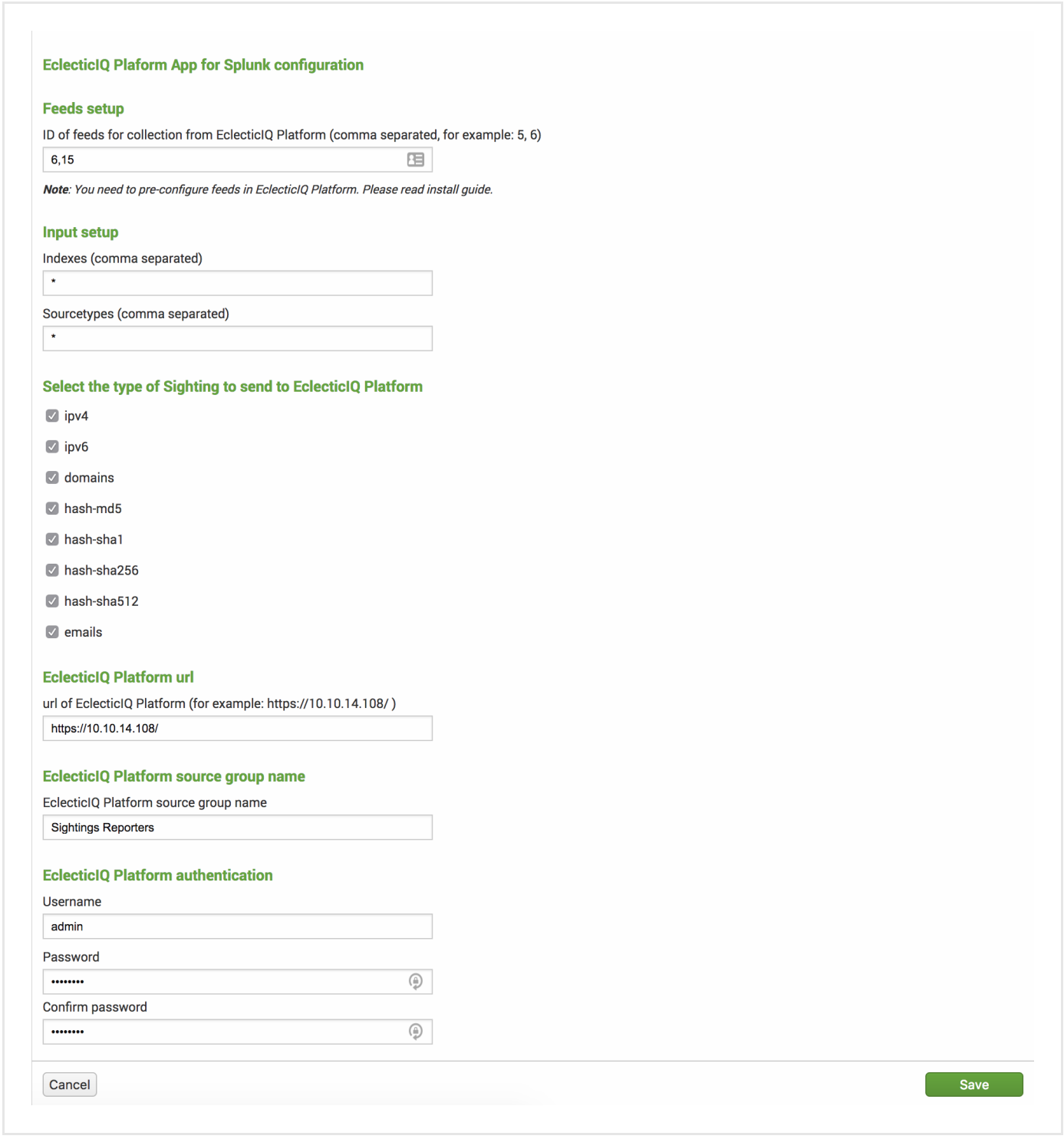
Install the app

- In the Splunk management console go to **Apps > Manage Apps**, and then click **Install app from file**.
- Browse to the location where the *eclecticiq-platform-app-for-splunk-<version_number>.tgz* file is stored, and then click **Upload**.
- After successfully completing the upload and the installation, restart Splunk.

Configure the app

After restarting Splunk, you can proceed to configuring EclecticIQ Platform App for Splunk.

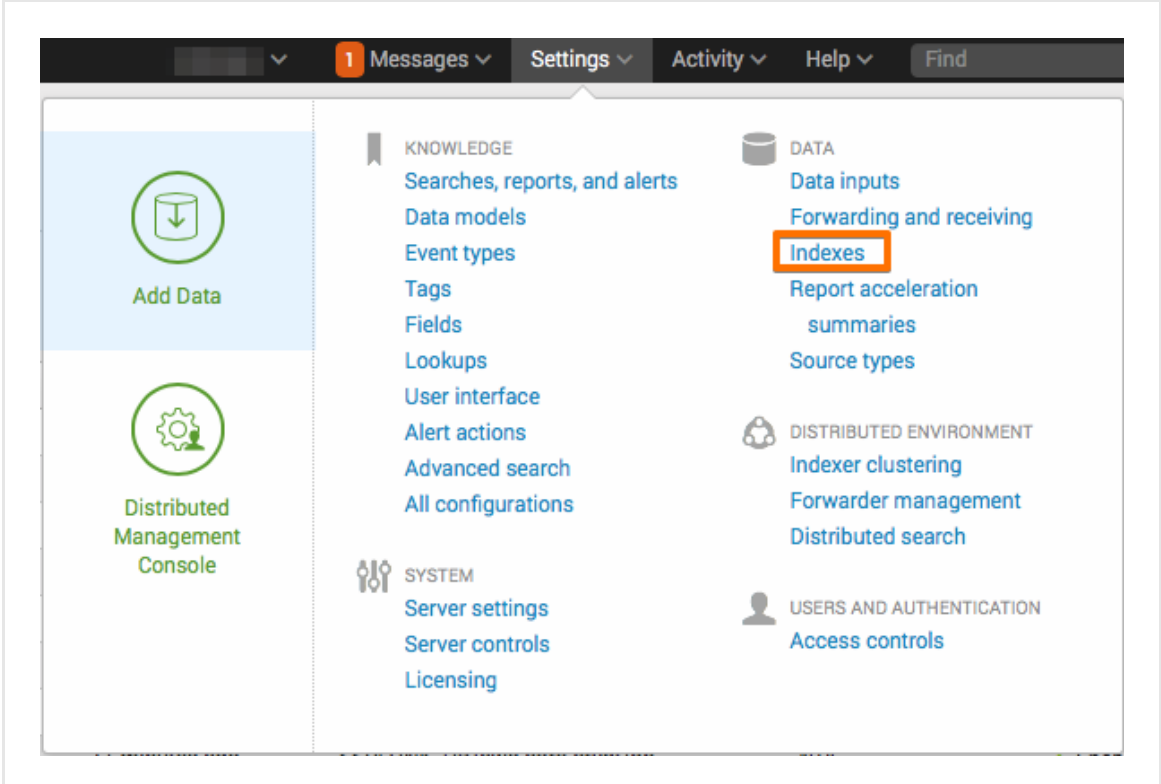
- In the Splunk management console go to **Apps**.
- From the app list select **EclecticIQ Platform App for Splunk**.
- On the displayed dialog window click **Continue to app setup page**.



On the EclecticIQ Platform App for Splunk configuration screen, define the following options:

- **Feeds setup**: enter the feed ID of the EclecticIQ Platform outgoing feeds whose content you want to send to Splunk. If you enter multiple feed IDs, use a comma (,) as a separator.
  Example: *4,18,74,88*

- **Input setup**: define the indexes and the source types you are using as data sources for this integration:
  - **Indexes**: enter the name of the **Splunk indexes** (`http://docs.splunk.com/splexicon:index`) you want to include as sources.
    Events included in the specified input indexes are searched for matches against the criteria defined in this configuration.
    Matching events are used to create sightings.
    If you enter multiple indexes, use a comma (,) as a separator.
    To view a list with the available Splunk indexes, in Splunk go to **Settings > Indexes**.
    Default value: * (asterisk, that is, all available Splunk indexes are included as sources)
  - **Sourcetypes**: enter the name of the **Splunk source types** (`https://docs.splunk.com/splexicon:sourcetype`) you want to include.
    Events whose data structure corresponds to the specified input source types are searched for matches against the criteria defined in this configuration.
    Matching events are used to create sightings.
    If you enter multiple source type names, use a comma (,) as a separator.
    Splunk includes a **built-in source type set** (`http://docs.splunk.com/documentation/splunk/latest/data/listofpretrainedsourcetypes`).
    Default value: * (asterisk, that is, all available source types are included as sources)
    Example: *access_combined,linux_messages_syslog*



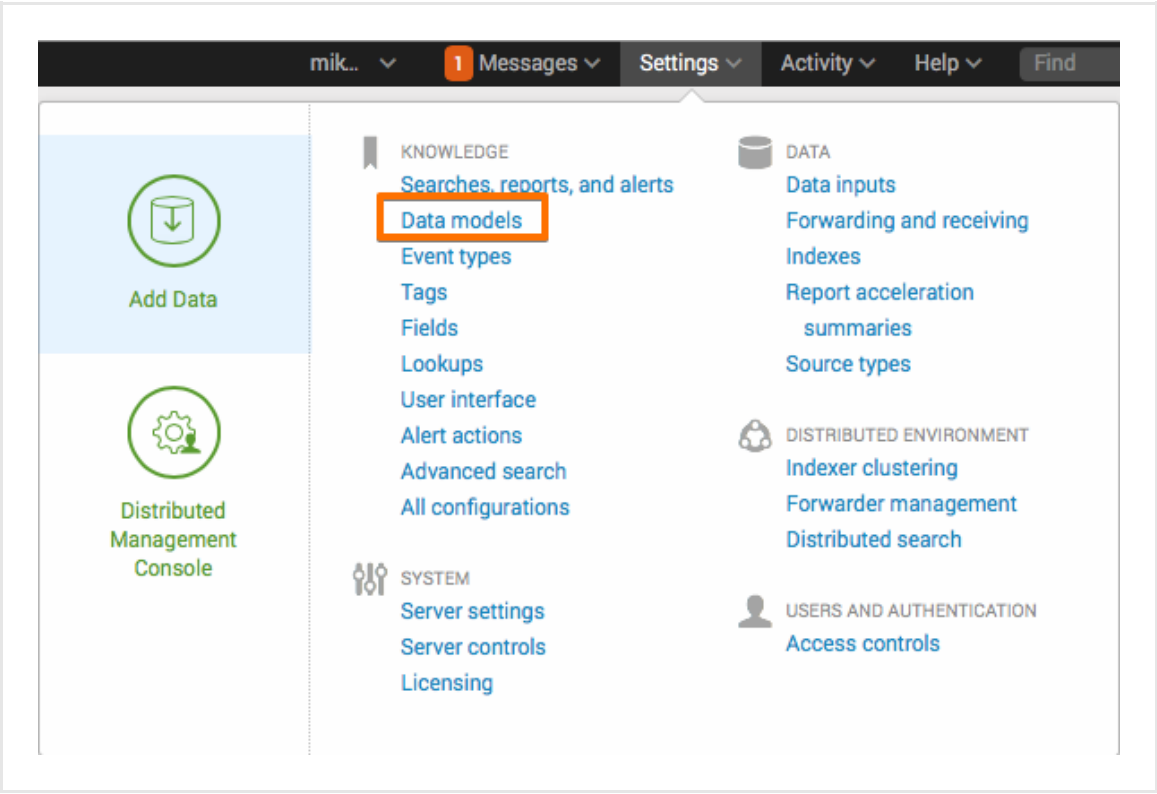
- **Select the type of Sighting to send to EclecticIQ Platform**: select all applicable checkboxes corresponding to the data types you want to use to generate the sightings that are subsequently sent for ingestion to EclecticIQ Platform. Supported types:
  - *ipv4*
  - *ipv6*
  - *domains*
  - *hash-md5*
  - *hash-1*
  - *hash-256*
  - *hash-512*
  - *email*
- **EclecticIQ platform URL**: enter the URL corresponding to the address of the EclecticIQ Platform host.
  Example: *https://10.10.10.10/ or https://platform.instance.org/*
- **EclecticIQ source group name**: enter the name of the group you want to use as a source.
  A valid group name corresponds to the name of any available group configured in the platform.
  Example: *Sightingbusters*
- **EclecticIQ platform authentication**: enter valid credentials to authenticate and to sign in to the platform; that is, a valid user name and a password, which you need to confirm.
- Click **Save** to save and store your configuration.

Configure data model acceleration

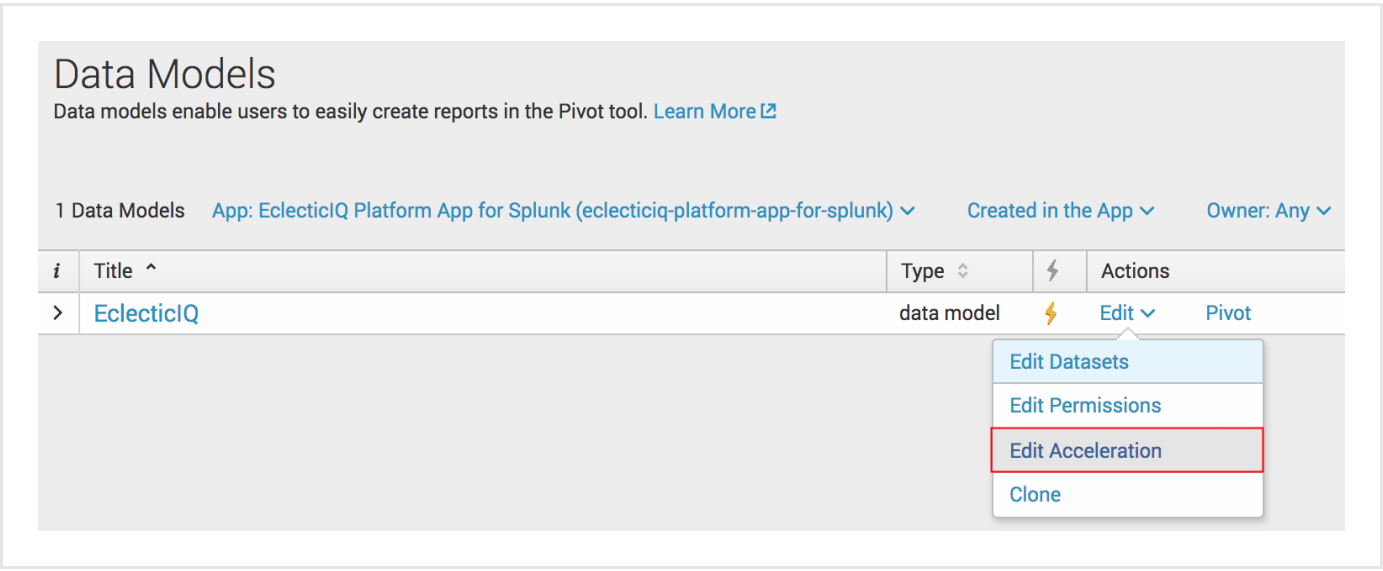
By default, **data model acceleration** `(https://docs.splunk.com/documentation/splunk/latest/knowledge/acceleratedatamodels)` is configured to speed up data models within 7 days.

To modify the data model acceleration settings, do the following:

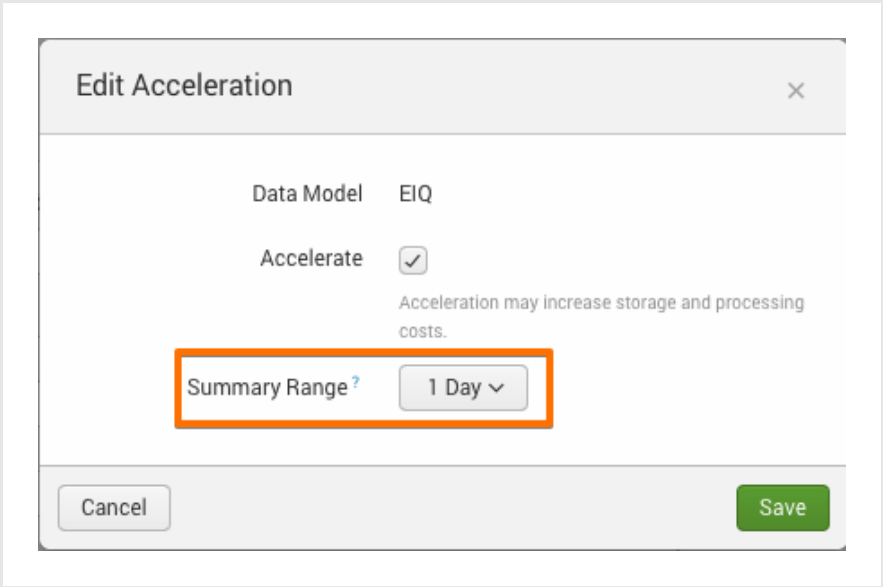
- In Splunk, go to **Settings > Data models**.



- Browse to the **EclecticIQ** row, and then select the **Edit > Edit Acceleration** menu option.



- In the displayed dialog window, make sure the **Accelerate** checkbox is selected.
- From the **Summary Range** drop-down menu, select the time interval you want data to base acceleration on.



- Click **Save** to save and store your edits.

Default job schedule

- By default, a script is configured to run and collect outgoing feeds once every 2 hours at *hour:00 mins*; that is, at 00:00, 02:00, 04:00, and so on.
- By default, a script is configured to push sightings once a day at 01:00 AM.

Customize the job schedule

You can change the job schedules in the following configuration file:
$SPLUNK_HOME/etc/apps/eclecticiq-platform-app-for-splunk/default/inputs.conf

This is the default version of the file that ships with the app:

```
[default]

[script://$SPLUNK_HOME/etc/apps/eclecticiq-platform-app-for-splunk/bin/eiq_send_sightings.py]
disabled = false
interval = 00 01 * * *

[script://$SPLUNK_HOME/etc/apps/eclecticiq-platform-app-for-splunk/bin/eiq_collect_feeds.py]
disabled = false
interval = * */2 * * *

[script://$SPLUNK_HOME/etc/apps/eclecticiq-platform-app-for-splunk/bin/eiq_setup_handler.py]
passAuth = splunk-system-user

[script://$SPLUNK_HOME/etc/apps/eclecticiq-platform-app-for-splunk/bin/eiq_collect_feeds.py]
passAuth = splunk-system-user

[script://$SPLUNK_HOME/etc/apps/eclecticiq-platform-app-for-splunk/bin/eiq_send_sightings.py]
passAuth = splunk-system-user
```

eiq_collect_feeds.py is the script that collects outgoing feed data from EclecticIQ Platform.
To change the script execution schedule, edit the corresponding `interval` cron expression.

eiq_send_sightings.py is the script that sends sightings to EclecticIQ Platform.
To change the script execution schedule, edit the corresponding `interval` cron expression.

For further details on Splunk cron expressions, see the official **Splunk documentation on cron expressions** `(http://docs.splunk.com/documentation/splunk/latest/alert/definescheduledalerts#using_cron_expressions` and their **answers to common questions on cron expressions** `(https://answers.splunk.com/answers/120603/cron-expression-in-splunk.html).`

After correctly configuring EclecticIQ Platform App for Splunk to integrate and work with Splunk, the corresponding dashboard view should become populated with relevant results.

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 like Splunk and IBM QRadar.

The 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 our hands dirty, let's 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.
- Edit the ***setup.py*** `(https://github.com/eclecticiq/platform-extensions/blob/master/eclecticiq-extension-example/setup.py)` file.
- Import dependencies.
- Create a JSON schema for the UI, if necessary.
- Set a schema definition for validation.
- Create an enricher class that extends the `EnricherBase`class, where you configure the enricher behavior:
  - Assign specific Celery tasks the enricher should execute;
  - Assign the extract types you want the enricher to look for and retrieve.
- Restart Supervisor, so that all managed processes can configure the newly added extension in **Enrichment > Catalog**.
- 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, we make a boilerplate available: *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 personalize 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",
    packages=find_packages(),
    install_requires=[
        'intelworks-platform',
        'requests'
    ],
    include_package_data=True,
    entry_points={
        'eiq.extensions': [
            'example = eclecticiq.extensions.example:prepare_extension'
        ],
    }
)
```

Edit the setup file

These are the *setup.py* parts you can, and should, edit as applicable:

In the extension `name`, change `example` to a more meaningful name for your enricher extension, but leave the `eclecticiq-extension-` prefix as is.
Example: `eclecticiq-extension-fraud-ip-observables`

Boilerplate:

```
name='eclecticiq-extension-example',
```

Example:

```
name='eclecticiq-extension-fraud-ip-observables',
```

Change the `version` number as appropriate.
Example: `1.1`

Boilerplate:

```
version="1.0",
```

Example:

```
version="1.1",
```

Change the `description` value, so that it provides basic details about the enricher extension.
Example: `Custom extension to retrieve observables on fraudulent IPs`

Boilerplate:

```
description="Example extension for EclecticIQ Platform",
```

Example:

```
description="Custom extension to retrieve observables on fraudulent IPs",
```

Add to the `install_requires` list the Python libraries and modules the extension needs to access for it to work as expected.
The Python libraries and modules you need to import and include in this list vary, depending on the extension you are building.

Boilerplate:

```
install_requires=[
    'eiq-platform',
    'requests'
],
```

Example:

```
install_requires=[
    'eiq-platform',
    'requests',
    'cabby',
    'furl'
],
```

`eiq.extensions` is the defined 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`

Boilerplate:

```
'eiq.extensions': [
    'example = eclecticiq.extensions.example:prepare_extension'
],
```

Example:

```
'eiq.extensions': [
    'fraud-ip = eclecticiq.extensions.fraud-ip-observables:prepare_extension'
],
```

Edit the init file

This part of the procedure customizes the fixtures you will need to run later to initialize the extension, after enabling it. Let's do it now before we forget.

The functions that take care of initializing the extension you are building are `prepare_extension` and `create_fixtures`. In `prepare_extension` you specify what the function should get ready, and in `create_fixtures` you define what the function should configure and set the before you execute the extension for the first time.

In this file, change the `Extension` return values and the enricher `params` metadata as applicable. Use the actual, correct names, descriptions, and values you define, set, and plan to use in your extension.

Open `/eclecticiq/extensions/example/__init__.py`.
First, import your custom extension, so that we actually have something to initialize.

Boilerplate:

```
from .enrichers import enrich_from_social_network
```

Example:

```
from .enrichers import FraudIPExtension
```

Inside the `prepare_extension` function, change the `description` and `enrichers` metadata values `Extension` returns, so that they reflect the actual values you use in your extension.

Boilerplate:

```
def prepare_extension():
    return Extension(
        name=__name__,
        description='Example extension for EclecticIQ Platform',
        enrichers=[
            enrich_from_social_network
        ]
    )
```

Example:

```
def prepare_extension():
    return Extension(
        name=__name__,
        description='Custom extension to retrieve observables on fraudulent IPs',
        enrichers=[
            FraudIPExtension
        ]
    )
```

Inside the `create_fixtures` function, assign a unique alphanumeric identifier value to `uuid`, and then proceed to edit the `params` metadata by replacing the boilerplate values with the actual types, values, field names, and flags defined in your enricher extension.

Boilerplate:

```
params={
            # Enricher name and description
            'name': 'Example enricher from social networks',
            'description': 'Query for registered Twitter/Facebook '
                           'accounts with provided handle/name',

            # Mapping enricher model to enricher task name
            'task_name': enrich_from_social_network.name,

            # Default values for enricher task parameters.
            # Types must match the ones configured in UI and marshmallow schema
            'parameters': {
                'check_twitter': True,
                'check_facebook': True
            },

            # Should the enricher be active by default
            'is_active': True,

            # Observable types supported as inputs for the enricher
            'input_extract_types': [
                'handle', 'name', 'person'],

            # If enricher creates entities, this is the reliability
            # that will be assigned to them.
            'source_reliability': 'C',

            # URL templates to original data from enricher source
            'source_urls': {
                'handle': 'https://twitter.com/${input}',
                'name': 'https://twitter.com/${input}',
                'person': 'https://twitter.com/${input}',
            }
        },
```

Example:

```
params={
            # Actual name and description
            # you defined for your enricher extension
            'name': 'Fraud IP extension',
            'description': 'Query XYZ intel provider to retrieve'
                           'IP and whois info with provided ip/domain name',

            # Map the enricher model to the enricher task name
            'task_name': FraudIPExtension.name,

            # Default values for the enricher task parameters.
            # Types must match the ones configured in UI and Marshmallow schemas
            'parameters': {
                'check_ip': True,
                'check_domain': True
            },

            # Default enricher status: either enabled or disabled
            'is_active': True,

            # Observable types supported as inputs for the enricher
            'input_extract_types': [
                'ipv4', 'ipv6', 'domain'],

            # If the enricher creates entities, this is the
            # reliability value assigned to them.
            'source_reliability': 'C',

            # URL templates to original data from enricher source
            # URL structure must match URLs pointing to original data
            'source_urls': {
                'ipv4': 'https://<example.com>/${input}',
                'ipv6': 'https://<example.com>/${input}',
                'domain': 'https://<example.com>/${input}',
            }
        },
```

Import dependencies

Make sure you include in `eclecticiq/extensions/example/enrichers.py` the necessary Python libraries and modules, so that the 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
`import requests`	Adds handy automation to **HTTP requests** (`http://docs.python-requests.org/en/master/`).
`from furl import furl`	Simplifies **URL manipulation** (`https://github.com/gruns/furl`).
`from marshmallow import Schema, fields`	Marshmallow schemas are used to validate UI schemas and form input.

Dependency	Description
`from eiq.platform.taskrunner.enricher import (EnricherBase, EnrichmentResult)`	The `EnricherBase` and `EnrichmentResult` classes help you define the enricher extension behavior, and how the enrichment extract results are stored and output.

Include the UI schema

You can skip this section if your enricher extension does not include UI components.

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.

Create the UI schema

You include the UI schema inside the enricher class.
The enricher class extends the `EnricherBase` class.
Example:

```
class FraudIPExtension(EnricherBase):

    # Internal unique task name of an enricher.
    # MUST keep the "eiq.enrichers." prefix
    name = 'eiq.enrichers.fraud_ip_extension'
```

Include your UI schema in the enricher class as a JSON array.
Example:

```
    # Definition of the UI form rendered in the platform UI:
    # Enrichment > Catalog > Edit enricher task form
    # "ui_form_schema" is the UI schema name; do not change it.
    ui_form_schema = [

        {
            "label": "Check IP",
            "name": "check_ip",
            "required": True,
            "type": "checkbox",
            "format": "bool",
            "hint": "Enable or disable IP lookup"
        },

        {
            "label": "Check domain name",
            "name": "check_domain",
            "required": True,
            "type": "checkbox",
            "format": "bool",
            "hint": "Enable or disable domain name lookup"
        },
    ]
```

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.
- You define each field using key/value pairs.
- Each key/value pair describes an attribute of the field.

Example:

```
[
  {_field_1_}
  {_field_2_}
...
]
```

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 `/api/extensions/` endpoint.
For example: *includeWhois*

`label`

The name of the field as displayed as a label on the resulting object in the UI form.
For 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 the text input format.
    Allowed values:
    - `datetime`
    - `host`
    - `url`
    - `email`
    - `regex`
    - `path`
    - `text`
    - `int`
    - `float`
    - `bool`
- `textarea`: a multiple row text input field.
- `password`: an 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 only 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.
For 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.

Set the schema definition

You can skip this section if you do not need a UI JSON schema for your enricher extension.

If you include a UI JSON schema, you also need to specify the schema definition you are going to validate the UI schema against.
The schema definition used to validate UI schema and form input is based on a Marshmallow schema definition.
The **Marshmallow schema** `(https://marshmallow.readthedocs.io/en/latest/)` defines the behavior of the controls and components on the UI form, and the `ui_form_schema` JSON schema needs to match it to pass validation.

First, make sure you import the following classes from Marshmallow:

```
from marshmallow import Schema, fields
```

Then, define your schema definition to validate the `ui_form_schema` JSON schema against.
Example:

```
class FraudIPExtensionSchema(Schema):
    check_ip = fields.Boolean(required=True)
    check_domain = fields.Boolean(required=True)
```

Lastly, include `parameters_schema` in your extension enricher class, and set it so that it points to the appropriate schema definition for the UI schema validation.
Example:

```
    # Schema used for validation and de-serialization of the enricher parameters
    parameters_schema = FraudIPExtensionSchema()
```

Define the enricher behavior

Your enricher class extends `EnricherBase`, so that your enricher is associated to a set of Celery tasks. Inside the enricher class, you define a unique name for the Celery task you want to associate to the enricher. The naming format for enricher Celery tasks is `eiq.enrichers.<your_extension_name>`.
Example:

```
class FraudIPExtension(EnricherBase):

    # Internal unique task name of an enricher.
    # MUST keep the "eiq.enrichers." prefix
    name = 'eiq.enrichers.fraud_ip_extension'
```

Now you need to map the enricher to the desired observable types you want it to handle as input data, and you need to configure it so that it outputs relevant observables.
The `enrich` function is the workhorse you need to do all the grunt work:

```
def enrich(self, entity, inputs):
```

In the `enrich` function you can define the logic to search for and retrieve the desired input data, any conditional flow and error handling, and how to output the input data as valid observables for the platform.
Example:

```
# Work method of an enricher
   def enrich(self, entity, inputs):

       # Get the parameters stored in the enricher task
       parameters = self.request.platform_task.parameters
       # If necessary, add here any input params users need
       # to input, for ex. in the UI, to config the enricher task
       # Example:
       param_name1 = param_value["args"]
       param_name2 = "param_value"

       # Keep the raw response headers for bookkeeping
       raw_responses = []

       extracts = []

       for extract in inputs:

         ...
         # Here your magic happens :)
         ...
```

The `EnrichmentResult` class helps store and handle output data. It returns the following output:

- Raw response, that is, enrichment raw data
- Observables (as a list)
- Entities (as list of entity IDs),

When building a custom enricher, 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 returned.

```
return EnrichmentResult(
    raw_data=json.dumps(raw_responses).encode('utf-8'),
    extracts=extracts,
    entities=[])
```

Package and deploy the extension

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.
- Launch the platform Python virtual environment. To enable `venv`, run the following command(s):

```
$ . /opt/eclecticiq/platform/api/bin/activate
```

or:

```
$ source /opt/eclecticiq/platform/api/bin/activate
```

- In the virtual environment, install the extension by running the following command(s):

```
$ pip install /tmp/eclecticiq-extension-example-1.0.tar.gz
```

The `pip` example installs from `/tmp/` to avoid dealing with file access rights and permissions.

Restart the processes

After completing the extension installation restart all *Supervisor* processes, so that all managed processes can configure the newly added extension in **Enrichment > Catalog**.

- Reload Supervisor configurations and restart all Supervisor-managed processes by running the following command(s):

```
$ supervisorctl reload
```

or:

```
$ supervisorctl reload all
```

- To restart all Supervisor-managed processes without reloading the supervisor configuration, run the following command(s):

```
$ supervisorctl restart
```

Check that the extension 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
set -e

readonly HTTPIE=http
readonly HTTPIE_ARGS="--check-status --verify=no"
readonly USERNAME=test
readonly PASSWORD=test

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
`https://<platform_host>/`	*Required* — The name of the host used to reach the API endpoint and to communicate with the API service.
`POST`, `GET`, `PUT`, `DELETE`	*Required* — A valid **HTTP method** `(http://www.restapitutorial.com/lessons/httpmethods.html)` to create, read, update, or delete a resource.
`/<API_endpoint>/`	*Required* — A relative URL pointing to the API endpoint that exposes the service you want to consume.
`?url=true&query=search-or-filter&params=4`	*Optional* — 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 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 extension by name, description, or creation date.
If your extension is included in the returned list, it is registered correctly.

Enable the extension

- 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.com 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" : { ... } }`

Initialize the extension

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):

```
# run this command to login as root with current user/pw
$ sudo su -

# run this command to login as root with eclecticiq user/pw
$ su - eclecticiq

# run this command to login as root with elasticsearch user/pw
$ su - elasticsearch

# run this command to login as root with logstash user/pw
$ su - logstash

# run this command to login as root with neo4j user/pw
$ su - neo4j

# run this command to login as root with nginx user/pw
$ su - nginx

# run this command to login as root with postgres user/pw
$ su - postgres

# run this command to login as root with redis user/pw
$ su - redis
```

- Explicitly set the platform environment variable in the platform configuration file:

```
$ export EIQ_PLATFORM_SETTINGS=/opt/eclecticiq/etc/eclecticiq/platform_settings.py
```

- Launch the platform Python virtual environment. To enable `venv`, run the following command(s):

```
$ . /opt/eclecticiq/platform/api/bin/activate
```

or:

```
$ source /opt/eclecticiq/platform/api/bin/activate
```

- Start a Python shell:

```
$ /opt/eclecticiq/platform/api/bin/manage 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 extension

“Nah, my code doesn’t need testing.”
(anonymous, got fired)

Test the extension 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.

Make sure you import the libraries, modules and classes you need to test your extension. For a standard basic enricher, this is what you typically need:

```
from collections import namedtuple

import httpretty, json

from <path.to.your.custom.extension> import <CustomExtensionName>
```

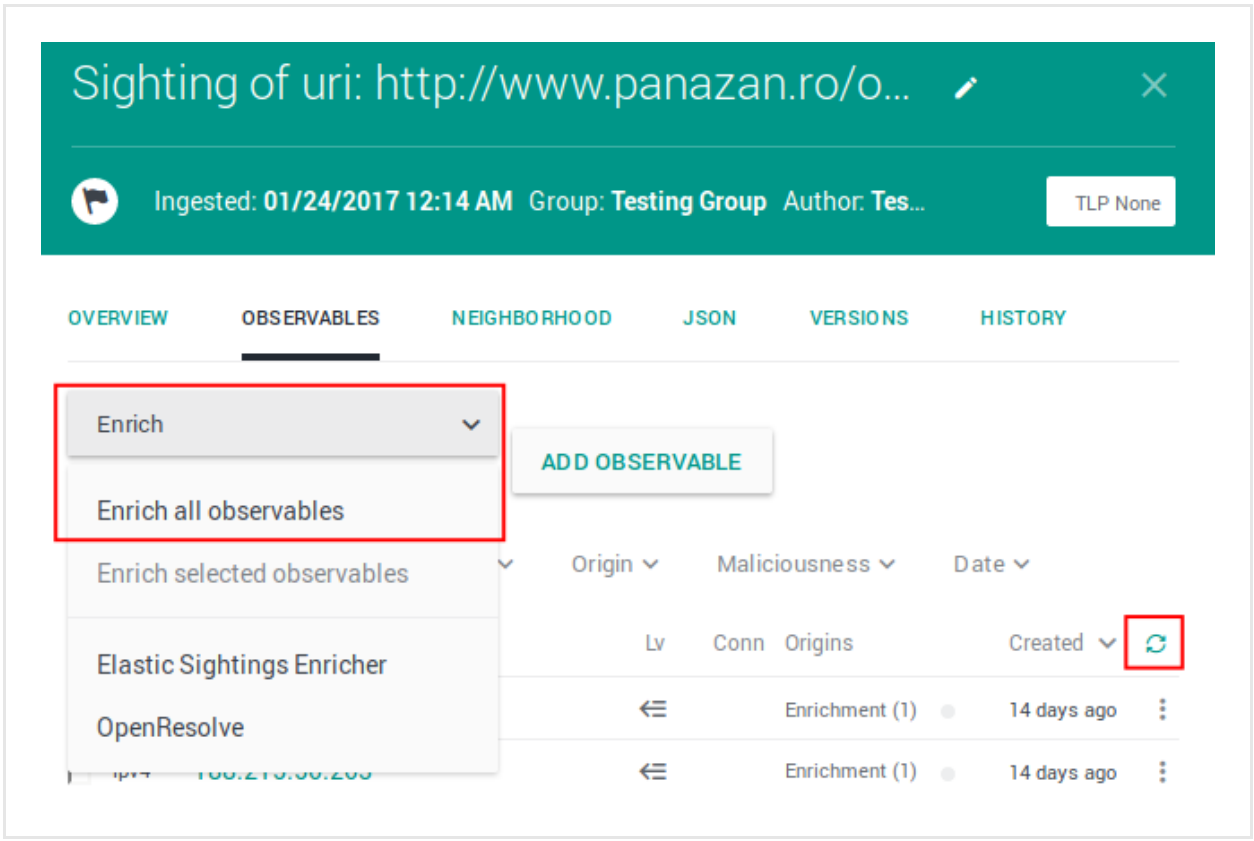
```
# Declare the task as a named tuple
DummyTask = namedtuple('DummyTask', ['parameters'])


@httpretty.activate
# This example uses dummy names and values.
# Replace them with the appropriate ones for your extension.
# Declare the function to use to test your extension enricher
def test_FraudIPExtension():

    enricher = FraudIPExtension()

    # Celery is not available to set up tasks
    # so you need to set it up here
    enricher.request.update(platform_task=DummyTask(parameters={
        ...
        # Pass the actual params
        # configured in your extension
        ...
    }))

    httpretty.register_uri(

        # Mock the API endpoint and any additional URL params
        httpretty.GET, 'https://api.com/endpoint/<fraud_ip_address>',

        # Mock the HTTP status code in the reponse
        status=200,

        # Mock the body of the response
        body=json.dumps({"body_content": "true"}),

        # Mock the appropriate content type
        # for the reponse
        content_type="application/json"
    )

    result = enricher.enrich(None, [{

        # Mock the enricher type
        # Ex: ipv4, domain, host, etc.
        'kind': 'ipv4',

        # Mock the enricher value,
        # based on the enricher type
        'value': '<fraud_ip_address>'
    }])

    # Verify that the response returns
    # the expected amount of observables
    # generated from the retrieved data
    assert len(result.extracts) == 3
```

Test the extension through the platfom UI

- To check if your enricher extension is available in the platform UI, go to **Enrichment > Catalog**.
- Your custom enricher should displayed in the tiled overview, and the corresponding **Enabled** checkbox should be selected to notify it is enabled.
- To test if your enricher extension works as expected, look for an entity with observables that your enricher 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.

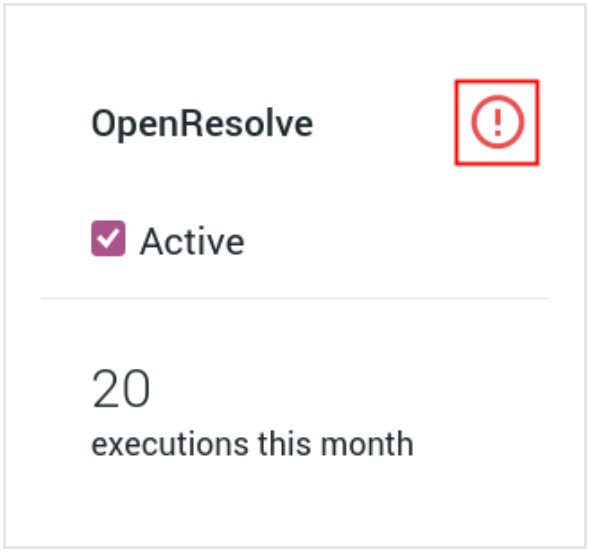
Alternatively:

- From the **Enrich** drop-down menu, select **Enrich all observables**.
- 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:
- Select it from the **Enrich** drop-down menu, and then click it.
- The platform polls the specified enricher for the entity, and it enriches all the entity observables with the retrieved data.

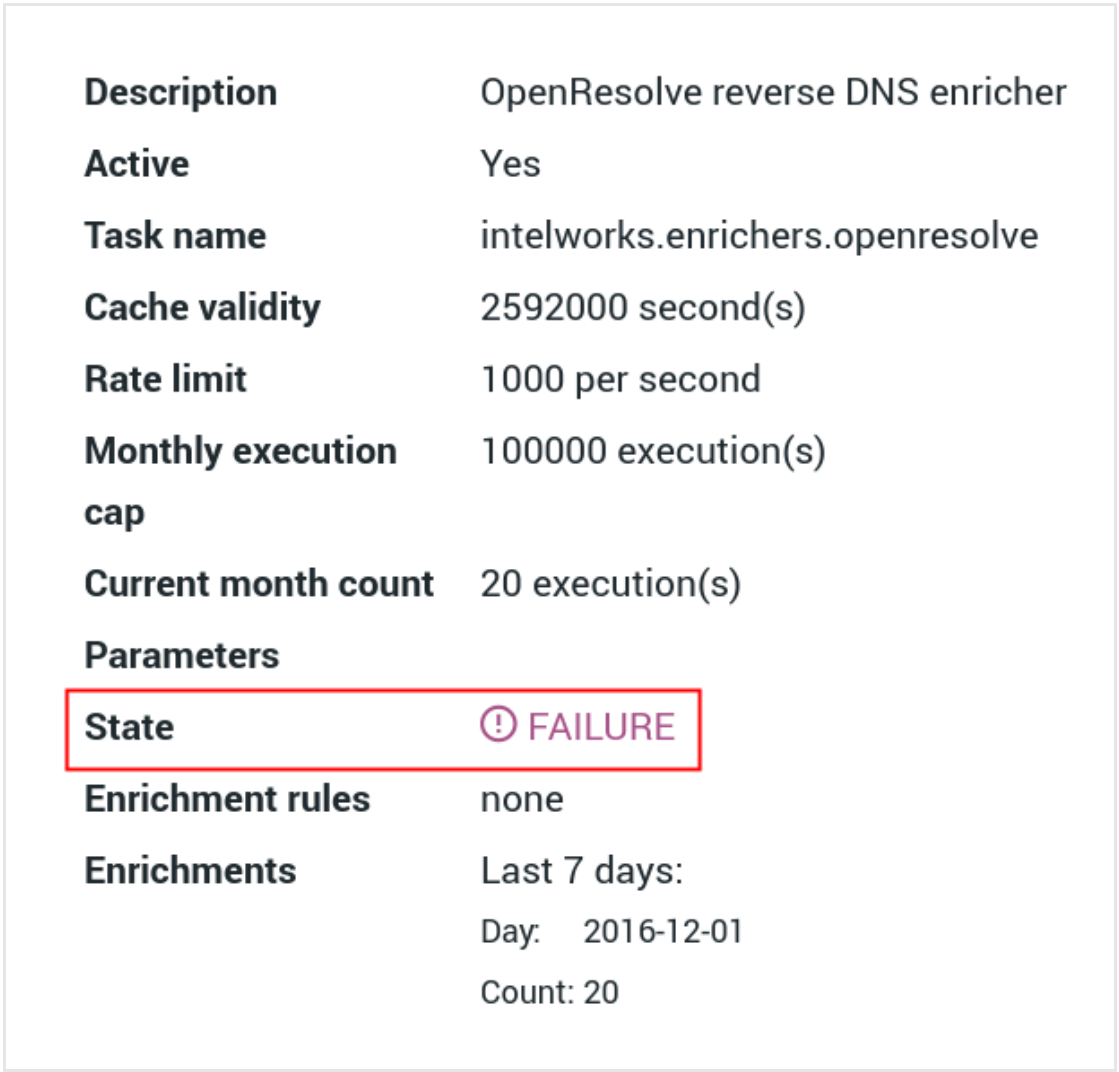


If you do not see any new observables after polling your enricher extension, check if the enricher crashed, and start investigating possible causes for the malfunction.

- In the platform UI, go to **Enrichment > Catalog**.
  On the **Enrichment > Catalog** tab you can see an 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.



- Click the enricher tile.
- On the enricher detail page, click **(!) Failure**.



- 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.