Runbook Enriching with Threat Intelligence Information

Date of Publish: 2018-11-15

Contents

Enriching Threat Intelligence Information

You can enrich your telemetry with threat intelligence information.

You can choose to skip this section and come back to it later if you don't want to apply threat intelligence at this time.

HCP provides an extensible framework to plug in threat intelligence sources. The threat intelligence feeds are loaded into a threat intelligence store similar to how the enrichment feeds are loaded. The keys are loaded in a key-value format. The key is the indicator and the value is the JSON formatted description of what the indicator is. When threat intelligence is applied as an enrichment to a field, HCP looks up the value of the field in the threat intelligence loaded into HBase. If the field value is found, HCP sets the is_alert field to true.

You can bulk load threat intelligence information from the following sources:

- CSV Ingestion
- HDFS through MapReduce
- Taxii Loader

We recommend using a threat feed aggregator such as Soltra to dedup and normalize the feeds via STIX/Taxii. HCP provides an adapter that is able to read Soltra-produced STIX/Taxii feeds and stream them into HBase. HCP additionally provides a flat file and STIX bulk loader that can normalize, dedup, and bulk load or poll threat intel data into HBase even without the use of a threat feed aggregator.

Bulk Loading CSV Threat Intelligence Information

Metron is designed to work with STIX/Taxii threat feeds, but can also be bulk loaded with threat data from a CSV file. Command-separated values (CSV) is a simple file format used to store tabular data, such as a spreadsheet or database. Files in the CSV format can be imported to and exported from programs that store data in tables.

The shell script $METRON_HOME/bin/flatfile_loader.sh reads data from the local disk and loads the threat intelligence data into an HBase table. This loader uses the special configuration parameter inputFormatHandler to specify how to consider the data. The two implementations are BY_LINE and org.apache.metron.dataloads.extractor.inputformat.WholeFileFormat. The default is BY_LINE, which makes sense for a list of CSVs in which each line indicates a unit of information to be imported. However, if you are importing a set of STIX documents, then you want each document to be considered as input to the Extractor.

OPTIONAL: Create a Mock CSV Threat Intel Feed Source

Hortonworks Cybersecurity Platform (HCP) is designed to work with STIX/Taxii threat feeds, but can also be bulk loaded with threat data from a CSV file. In this example, we create a mock CSV enrichment source. If your production environment, you will want to use a genuine enrichment source.
Similar to enrichments, we need to set up a data.csv file, the extractor config JSON, and the enrichment config JSON. For this example, we use a Zeus malware tracker list located here: https://zeustracker.abuse.ch/blocklist.php?download=domainblocklist.

Procedure

1. Log into the $HOST_WITH_ENRICHMENT_TAG as root.
2. Copy the contents from the Zeus malware tracker list link to a file called domainblocklist.csv.

```
curl -o domainblocklist.txt https://zeustracker.abuse.ch/blocklist.php?
download=domainblocklist
```

   Make sure you don't have an empty newline character as the last line of the CSV file, as that will result in a null pointer exception.

3. Similarly to enrichment we will need to process this feed into a CSV so we can bulk load it into HBase:

```
cat domainblocklist.txt | grep -v "^#" | grep -v "^$" | grep -v "^https" |
 awk '{print $1",abuse.ch"}' > domainblocklist.csv
```

Configure a CSV Extractor Configuration File

You use the extractor configuration file to bulk load the threat intelligence enrichment store into HBase.

Procedure

1. Log in as root to the host on which Metron is installed.

```
sudo -s $METRON_HOME
```

2. Determine the schema of the threat intelligence source.

   The schema of our mock enrichment source is:

```
domain | source
```

3. Create an extractor configuration file called threatintel_extractor_config_temp.json at $METRON_HOME/config and populate it with the threat intelligence source schema:

```
{
"config" : {
   "columns" : {
      "domain" : 0
      ,"source" : 1
   }
   ,"indicator_column" : "domain"
   ,"type" : "zeusList"
   ,"separator" : ","
  }
  ,"extractor" : "CSV"
}
```

4. Remove any non-ASCII invisible characters that might have been included if you copy and pasted:

   iconv -c -f utf-8 -t ascii threatintel_extractor_config_temp.json -o threatintel_extractor_config.json

Configure Element-to-Threat Intelligence Feed Mapping

You now need to configure what element of a tuple should be enriched with what enrichment type. This configuration is stored in ZooKeeper.

Procedure

1. Log in as root user to the host that has Metron installed.

```
sudo -s $METRON_HOME
```

2. Copy and paste the following into a file called threatintel_config_temp.json at $METRON_HOME/config.

```
{
  "zkQuorum" : "localhost:2181"
 ,"sensorToFieldList" : {
    "squid" : {
           "type" : "THREAT_INTEL"
          ,"fieldToEnrichmentTypes" : {
             "domand_without_subdomains" : [ "zeusList" ]
                                         }
             }
```

```
                             }
}
```

The threatintel_config_temp.json file causes HCP to look up the value of the field in the HBase enrichment. If the field is found, HCP adds the HBase field values as new enrichments to the event.

You must specify the following:

- The ZooKeeper quorum which holds the cluster configuration
- The mapping between the fields in the enriched documents and the enrichment types.

This configuration allows the ingestion tools to update ZooKeeper post-ingestion so that the enrichment topology can take advantage immediately of the new type.

3. Remove any non-ASCII invisible characters that might have been included if you copy and pasted:

```
iconv -c -f utf-8 -t ascii threatintel_config_temp.json -o
 threatintel_config.json
```

Run the Threat Intelligence Loader

Now that you have defined the threat intel source, threat intel extractor, and threat intel mapping configuration, you must run the loader to move the data from the enrichment source to the HCP enrichment store (HBase) and store the enrichment configuration in ZooKeeper.

Note: There is a special configuration parameter to the Extractor config that is only considered during this loader:

inputFormatHander	This specifies how to consider the data. The two implementations are BY_LINE and org.apache.metron.dataloads.extractor.inputformat.WholeFileFormat The default is BY_LINE, which makes sense for a list of CSVs where each line indicates a unit of information which can be imported. However, if you are importing a set of STIX documents, then you want each document to be considered as input to the Extractor.

Procedure

1. Log into $HOST_WITH_ENRICHMENT_TAG as root and navigate to $METRON_HOME/config.
2. Use the loader to move the enrichment source to the enrichment store in ZooKeeper:

```
$METRON_HOME/bin/flatfile_loader.sh -n threatintel_config.json
 -i domainblocklist_ref.csv -t threatintel -c t -e
 threatintel_extractor_config.json
```

This command modifies the Squid enrichment config in ZooKeeper to include the threat intel enrichment.

The parameters for the utility are as follows:

-b,--batchSize <SIZE>	The batch size to use for HBase puts
-c,--hbase)cf <CF>	HBase column family to ingest the data into. This must be column family t.
-e,--extractor_config <JSON_FILE>	JSON Document describing the extractor for this input data source
-h,--help	Generate Help screen
-i,--input <FILE>	The CSV File to load

-l,--log4j <FILE>	The log4j properties file to load
-m,--import_mode <MODE>	The Import mode to use: LOCAL,MR.Default: LOCAL
-n,--enrichment_config <JSON_FILE>	JSON Document describing the enrichment configuration details. This is used to associate an enrichment type with a field type in ZooKeeper.
-p,--threads <NUM_THREADS>	The number of threads to use when extracting data. The default is the number of cores of your machine.
-q,--quiet	Do not update progress
-t,--hbase_table <TABLE>	HBase table to ingest the data into.

The data is populated into an HBase table called enrichment.

3. Verify that the logs were properly ingested into HBase:

```
hbase shell
scan 'threatintel'
```

4. Verify that the ZooKeeper enrichment tag was properly populated:

```
$METRON_HOME/bin/zk_load_configs.sh -m DUMP -z $ZOOKEEPER_HOST:2181
```

You should see a configuration for the Squid sensor something like the following:

```
{
  "index" : "squid",
  "batchSize" : 1,
  "enrichment" : {
    "fieldMap" : {
      "hbaseThreatintel" : [ "ip_src_addr" ]
    },
    "fieldToTypeMap" : {
      "ip_src_addr" : [ "user" ]
    },
    "config" : { }
  },
  "enrichment" : {
    "fieldMap" : { },
    "fieldToTypeMap" : { },
    "config" : { },
    "triageConfig" : {
      "riskLevelRules" : { },
      "aggregator" : "MAX",
      "aggregationConfig" : { }
    }
  },
  "configuration" : { }
}
```

5. Generate some data by using the Squid client to execute requests:

   a) Use ssh to access the host for Squid.

   b) Start Squid and navigate to /var/log/squid:

```
ssh <Nifi Host>

sudo su -
systemctl start squid
cd /var/log/squid
```

```
tail -f access.log
```

c) Generate some data by entering the following:

```
squidclient http://www.cnn.com
```

6. Generate some data by using the Squid client to execute http requests:

```
squidclient http://www.actdhaka.com
```

Bulk Loading TAXII Threat Intelligence Information

You can enrich your telemetry with threat intelligence information.

You can choose to skip this section and come back to it later if you don't want to apply threat intelligence information at this time.

Metron provides an extensible framework to plug in threat intel sources. The threat intelligence feeds are loaded into a threat intelligence store similar to how the enrichment feeds are loaded. The keys are loaded in a key-value format. The key is the indicator and the value is the JSON formatted description of what the indicator is. When threat intelligence is applied as an enrichment to a field, HCP looks up the value of the field in the threat intelligence loaded into HBase. If the field value is found, HCP sets the is_alert field to true.

We recommend using a threat feed aggregator such as Soltra to dedup and normalize the feeds via STIX/Taxii. Metron provides an adapter that is able to read Soltra-produced STIX/Taxii feeds and stream them into HBase. HCP additionally provides a flat file and STIX bulk loader that can normalize, dedup, and bulk load or poll threat intel data into HBase even without the use of a threat feed aggregator.

Fetch Hail a TAXII Feeds

After you install your TAXII provider, you must fetch the latest Hail a TAXII feeds into the TAXII server. Hail a TAXII.com is a repository of Open Source Cyber Threat intelligence feeds in STIX format.

Before you begin

Set up your TAXII provider. Refer to your TAXII provider documentation for more information.

Procedure

1. Fetch the lastest Hail a TAXII feeds into the TAXII server:

```
service opentaxii sync <service-name> [YYYY-MM-DD]
For example:
service opentaxii sync guest.phishtank_com
service opentaxii sync guest.Abuse_ch 2016-08-01
```

Note: The date (YYYY-MM-DD) indicates the time from when the threat intel feeds is to be pulled. If not suffixed, then the sync command picks up feeds available for the current day.

2. Repeate Step 1 for all subscribed services.

Configure a TAXII Extractor Configuration File

After you fetch the latest OpenTAXII feeds to the OpenTAXII server, you must create an extractor configuration file to bulk load the threat intelligence enrichment store into HBase.

Procedure

1. Log in as root to the host on which Metron is installed.

   ```
   sudo -s $METRON_HOME
   ```

2. Determine the schema of the threat intelligence source.

   The schema of our mock enrichment source is domain|owner|registeredCountry|registeredTimestamp.
3. Create an extractor configuration file called threatintel_extractor_config_temp.json at $METRON_HOME/config and populate it with the threat intelligence source schema:

   ```
   {
   "config" : {
       "columns" : {
           "ip" : 0
       }
       ,"indicator_column" : "ip"
       ,"type" : "malicious_ip"
       ,"separator" : ","
     }
     ,"extractor" : "STIX"
   }
   ```

4. Remove any non-ASCII invisible characters that might have been included if you copy and pasted:

   iconv -c -f utf-8 -t ascii threatintel_extractor_config_temp.json -o threatintel_extractor_config.json

Configure a TAXII Connection Configuration File

In addition to the Extractor configuration file, this TAXII loader requires a configuration file describing the connection information to the TAXII server.

Procedure

1. Log in as root to the host on which Metron is installed.

   ```
   sudo -s $METRON_HOME
   ```

2. Create a connection configuration file called threatintel_connection_config_temp.json at $METRON_HOME/config and populate it with the threat intelligence source schema:

   ```
   {
     "endpoint" : "http://localhost:9000/services/discovery"
     ,"username" : "guest"
     ,"password" : "guest"
     ,"type" : "DISCOVER"
     ,"collection" : "guest.MalwareDomainList_Hostlist"
     ,"table" : "threatintel"
     ,"columnFamily" : "t"
     ,"allowedIndicatorTypes" : [ "domainname:FQDN", "address:IPV_4_ADDR" ]
   }
   ```

   where:

endpoint	The URL of the endpoint
type	POLL or DISCOVER depending on the endpoint.
collection	The Taxii collection to ingest
table	The HBase table to import into

columnFamily	The column family to import into
allowedIndicatorTypes	an array of acceptable threat intelligence types (see the "Enrichment Type Name" column of the Stix table above for the possibilities).

The parameters for the utility are as follows:

Short Code	Long Code	Is Required?	Description
-h		No	Generate the help screen/set of options
-e	--extractor_config	Yes	JSON Document describing the extractor for this input data source
-c	--taxii_connection_config	Yes	The JSON config file to configure the connection
-p	--time_between_polls	No	The time between polling the Taxii server in milliseconds. (default: 1 hour)
-b	--begin_time	No	Start time to poll the Taxii server (all data from that point will be gathered in the first pull). The format for the date is yyyy-MM-dd HH:mm:ss
-l	--log4j	No	The Log4j Properties to load
-n	--enrichment_config	No	The JSON document describing the enrichments to configure. Unlike other loaders, this is run first if specified.

3. Remove any non-ASCII invisible characters that might have been included if you copy and pasted:

   iconv -c -f utf-8 -t ascii threatintel_extractor_config_temp.json -o threatintel_extractor_config.json

Push Hailataxii Feeds to HBase

After you create the extractor configuration and connection configuration files, you can push the HailaTAXII feeds from the OpenTAXII server into HBase.

Procedure

1. Push the HailaTAXII feeds from the OpenTAXII server into HBase:

```
/usr/metron/<METRON_VERSION>/bin/threatintel_taxii_load.sh -b <START_TIME>
 -c /path/to/connection_config.json
  -e /path/to/extractor.json -p <TIME_INTERVAL_MSECS>
For example:
/usr/metron/0.2.0BETA/bin/threatintel_taxii_load.sh -b "2016-08-01
 00:00:00" -c ~/connection_config.json -e ~/extractor.json -p 10000
```

2. Remove any non-ASCII invisible characters that might have been included if you copy and pasted:

   iconv -c -f utf-8 -t ascii threatintel_extractor_config_temp.json -o threatintel_extractor_config.json

Verify Threat Intelligence Feeds in HBase

After you push the HailaTAXII feeds to HBASE, you should check HBase for the threat intelligence information.

Procedure

Query the HBase table to check for threat intelligence feeds:

```
echo "scan 'threatintel'" | hbase shell
```

Using Stellar Data to Transform Threat Intelligence Data

You can transform and filter threat intelligence data as it is loaded into HBase by using Stellar extractor properties in the extractor configuration file. This task is optional.

Procedure

1. Transform and filter the threat intelligence data as it is loaded into HBase by using the following Stellar extractor properties in the extractor_config.json file at $METRON_HOME/config:

Extractor Property	Description	Example
value_transform	Transforms fields defined in the columns mapping with Stellar transformations. New keys introduced in the transform are added to the key metadata.	`"value_transform" : { "domain" : "DOMAIN_REMOVE_TLD(domain)"`
value_filter	Allows additional filtering with Stellar predicates based on results from the value transformations. In the following example, records whose domain property is empty after removing the TLD are omitted.	`"value_filter" : "LENGTH(domain) > 0", "indicator_column" : "domain",`
indicator_transform	Transforms the indicator column independent of the value transformations. You can refer to the original indicator value by using indicator as the variable name, as shown in the following example. In addition, if you prefer to piggyback your transformations, you can refer to the variable domain, which allows your indicator transforms to inherit transformations done to this value during the value transformations.	`"indicator_transform" : { "indicator" : "DOMAIN_REMOVE_TLD(indicator)"`
indicator_filter	Allows additional filtering with Stellar predicates based on results from the value transformations. In the following example, records whose indicator value is empty after removing the TLD are omitted.	`"indicator_filter" : "LENGTH(indicator) > 0", "type" : "top_domains",`

The following example uses a CSV list of top domains as an enrichment and filters the value metadata, as well as the indicator column, with each of the supported Stellar expressions:

```
{
 "config" : {
 "zk_quorum" : "$ZOOKEEPER_HOST:2181",
 "columns" : {
 "rank" : 0,
 "domain" : 1
 },
 "value_transform" : {
 "domain" : "DOMAIN_REMOVE_TLD(domain)"
 },
 "value_filter" : "LENGTH(domain) > 0",
 "indicator_column" : "domain",
 "indicator_transform" : {
 "indicator" : "DOMAIN_REMOVE_TLD(indicator)"
 },
 "indicator_filter" : "LENGTH(indicator) > 0",
 "type" : "top_domains",
 "separator" : ","
 },
 "extractor" : "CSV"
 }
```

Running a file import with the above data and extractor configuration will result in the following two extracted data records:

Indicator	Type	Value
google	top_domains	{ "rank" : "1", "domain" : "google" }
yahoo	top_domains	{ "rank" : "2", "domain" : "yahoo" }

Note: The extractor_config.json file is not stored anywhere by the loader. This file is used once by the bulk loader to parse the enrichment dataset. If you would like to keep a copy of this file, be sure to save a copy to another location.

2. To access properties that reside in the global configuration file, provide a ZooKeeper quorum via the zk_quorum property. If the global configuration looks like "global_property" : "metron-ftw", enter the following to expand the value_transform:

```
"value_transform" : {
    "domain" : "DOMAIN_REMOVE_TLD(domain)",
     "a-new-prop" : "global_property"
 },
```

The resulting value data will look like the following:

Indicator	Type	Value
google	top_domains	{ "rank" : "1", "domain" : "google", "a-new-prop" : "metron-ftw" }
yahoo	top_domains	{ "rank" : "2", "domain" : "yahoo", "a-new-prop" : "metron-ftw" }

3. Remove any non-ASCII characters:

```
iconv -c -f utf-8 -t ascii extractor_config_temp.json -o
 extractor_config.json
```

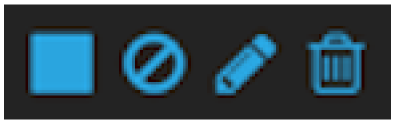
Map Fields to HBase Enrichments

Now that you have data flowing into the HBase table, you need to ensure that the enrichment topology can be used to enrich the data flowing past. You can refine the parser output through transformations, enrichments, and threat intelligence.
Each of the parser outputs is added or modified in the Schema field.

Procedure

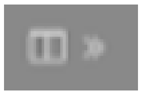
1. Select the new sensor from the list of sensors on the main window.

2. Click the pencil icon in the list of tool icons

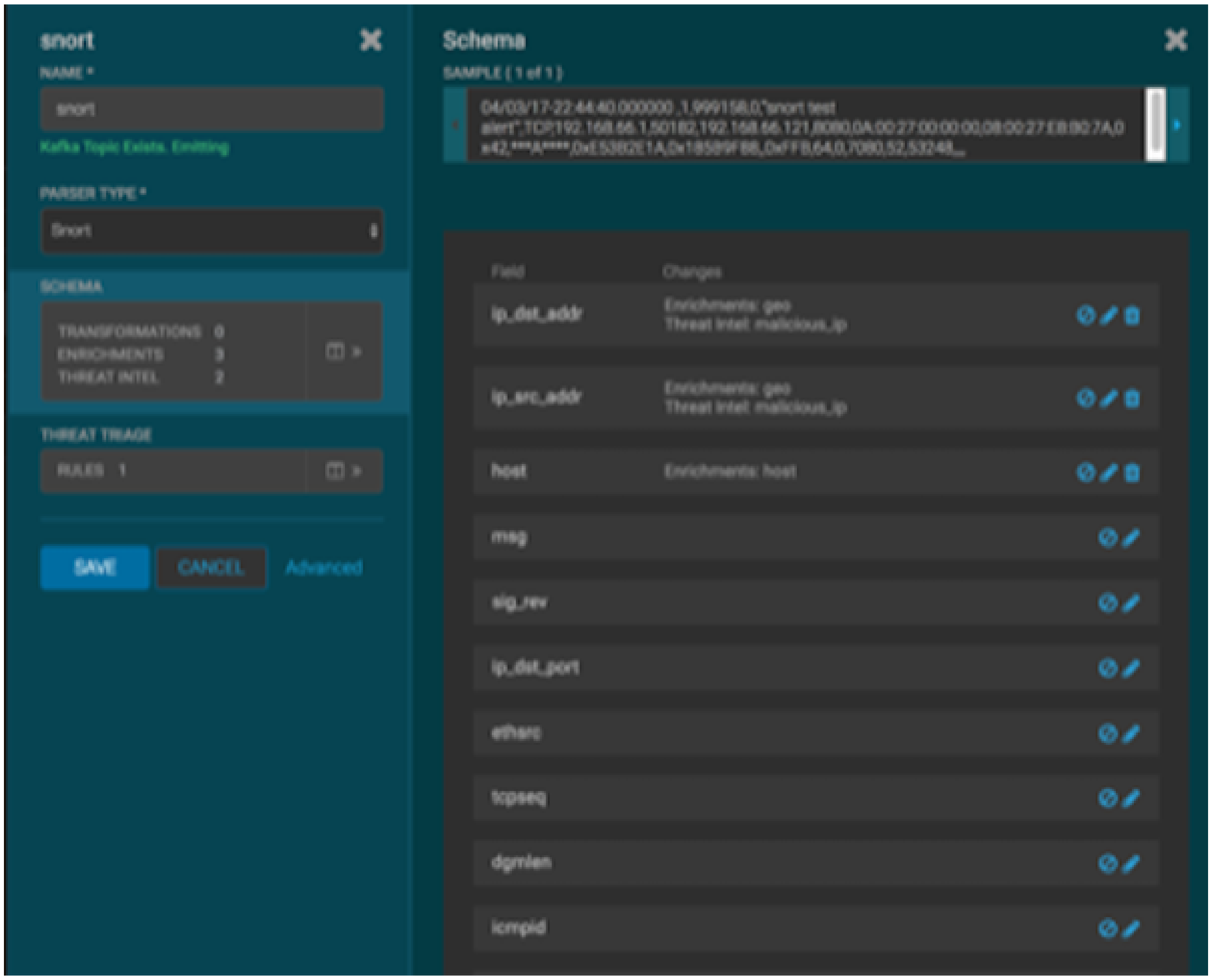
   for the new sensor.

   The Management Module displays the sensor panel for the new sensor.
3. In the Schema box, click

   (expand window button).

   The Management module displays a second panel and populates the panel with message, field, and value information.

   snort

   NAME *

   snort

   Kafka Topic Exists. Emitting

   PARSER TYPE *

   Snort

   SCHEMA

   TRANSFORMATIONS 0
   ENRICHMENTS 3
   THREAT INTEL 2

   THREAT TRIAGE

   RULES 1

   SAVE CANCEL Advanced

   Schema

   SAMPLE (1 of 1)

   ```
   04/03/17-22:44:40.000000 ,1,999158,0,"snort test
   alert",TCP,192.168.66.1,50182,192.168.66.121,8080,0A:00:27:00:00:00,08:00:27:E8:B0:7A,0
   x42,***A****,0xE5382E1A,0x18589F88,0xFFB,64,0,7080,52,53248,,
   ```

Field	Changes
ip_dst_addr	Enrichments: geo Threat Intel: malicious_ip
ip_src_addr	Enrichments: geo Threat Intel: malicious_ip
host	Enrichments: host
msg	
sig_rev	
ip_dst_port	
ethsrc	
tcpseq	
dgmlen	
icmpid	

   The Sample field, at the top of the panel, displays a parsed version of a sample message from the sensor. The Management module will test your transformations against these parsed messages.

   You can use the right and left arrow buttons in the Sample field to view the parsed version of each sample message available from the sensor.

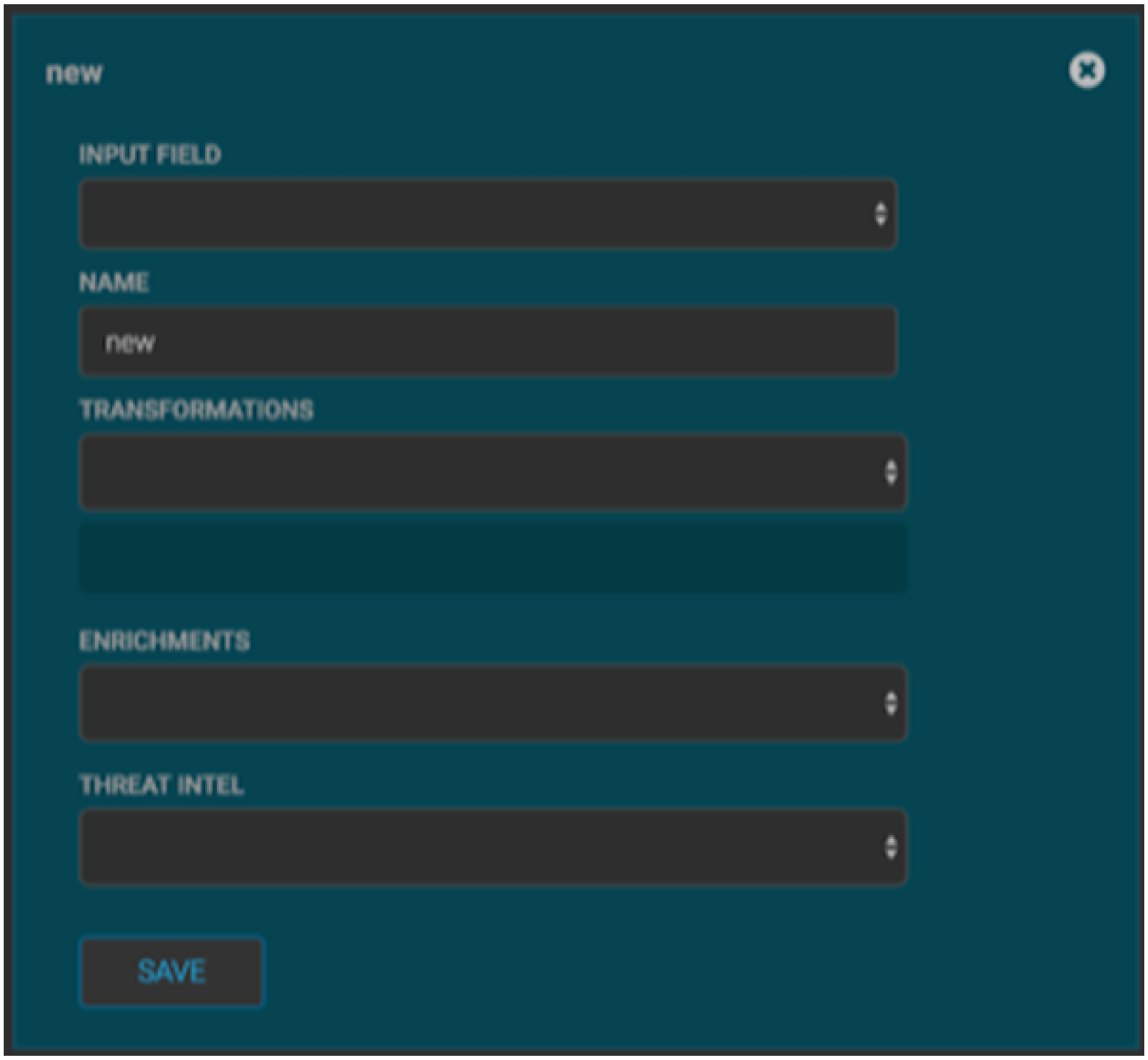
4. You can apply transformations to an existing field or create a new field. Click the

   (edit icon) next to a field to apply transformations to that field. Or click

   (plus sign) at the bottom of the Schema panel to create new fields.

   Typically users store transformations in a new field rather than overriding existing fields.

   For both options, the Management module expands the panel with a dialog box containing fields in which you can enter field information.



5. In the dialog box, enter the name of the new field in the **NAME** field, choose an input field from the **INPUT FIELD** box, and choose your transformation from the **TRANSFORMATIONS** field or enrichment from the **ENRICHMENTS** field.

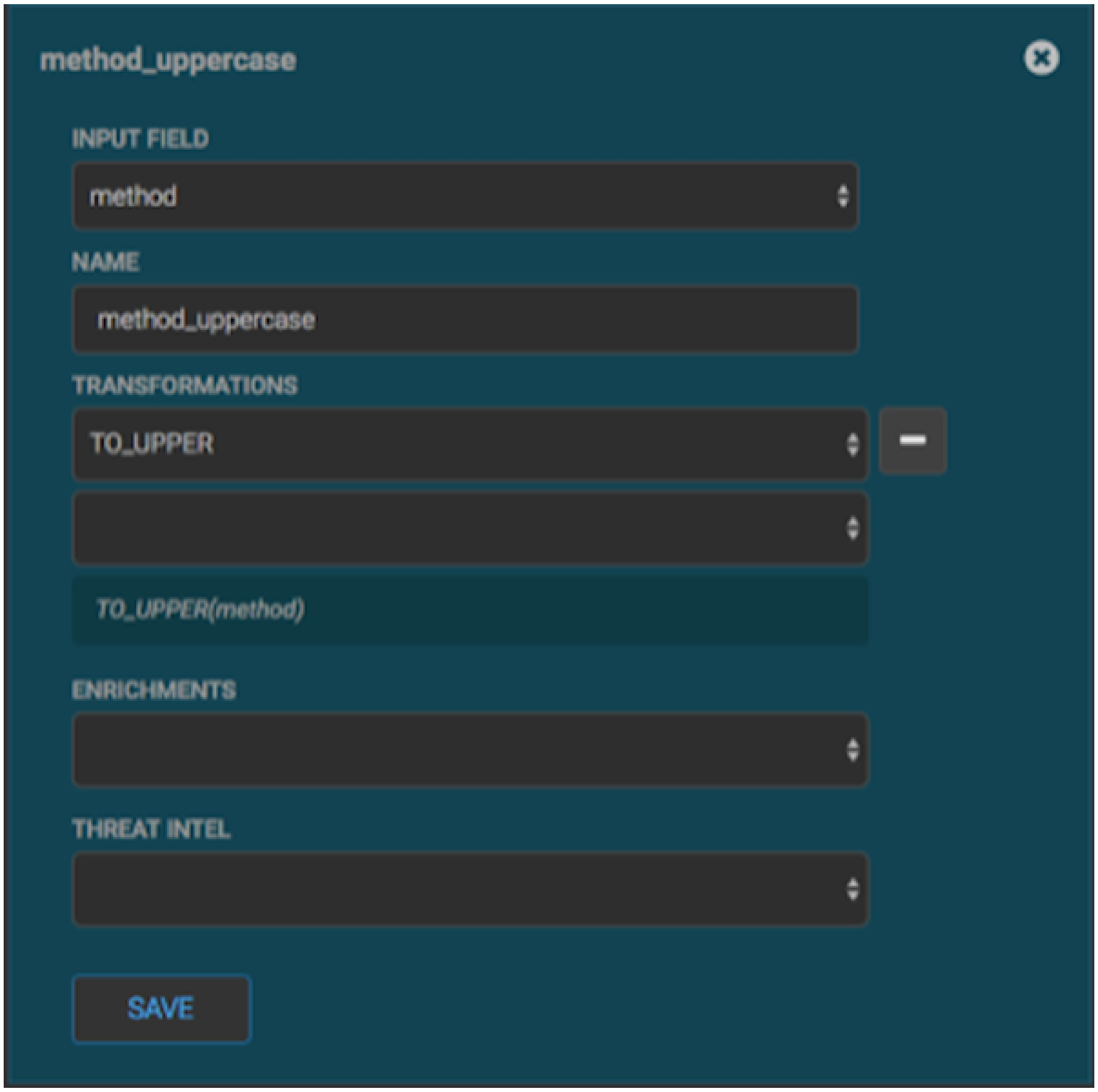
   For example, to create a new field showing the lower case version of the method field, do the following:

   a) Enter method-uppercase in the **NAME** field.

b) Choose method from the **INPUT FIELD**.

c) Choose TO_UPPER in the **TRANSFORMATIONS** field.

Your new schema information panel should look like this:



method_uppercase

INPUT FIELD: method

NAME: method_uppercase

TRANSFORMATIONS: TO_UPPER

TO_UPPER(method)

ENRICHMENTS

THREAT INTEL

6. Click SAVE to save your changes.
7. You can suppress fields from showing in the Index by clicking

   (suppress icon).
8. Click SAVE to save the changed information.

   The Management module updates the Schema field with the number of changes applied to the sensor.

Verify That the Threat Intel Events Are Enriched

After you finish enriching your new data source, you should verify that the output matches your enrichment information.

By convention, the index where the new messages are indexed is called squid_index_[timestamp] and the document type is squid_doc.

Procedure

From the Alerts UI, search the source:type filter for squid messages.