Understanding and Leveraging Cribl/CrowdStream and Next-Gen SIEM

Table of Contents

Simple platform definitions:

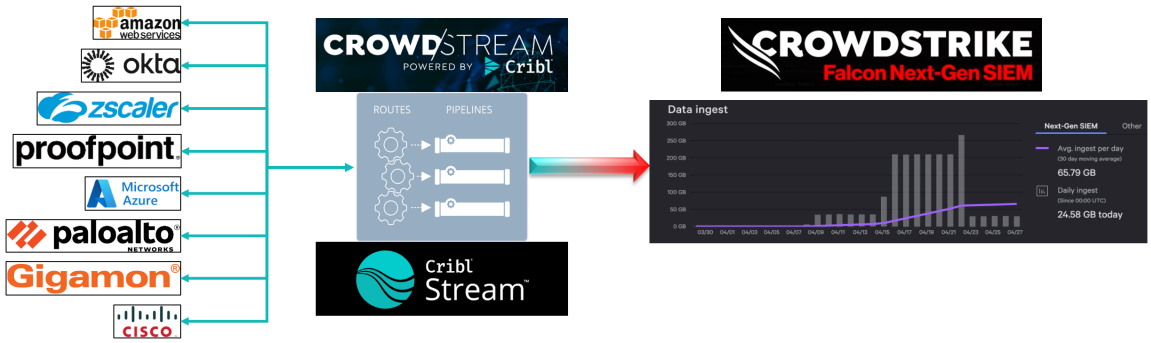
CrowdStrike Next-Gen SIEM: A cloud-native, petabyte-scale solution that is designed to provide unprecedented visibility across all of a customer's collected data.

Cribl Stream: A cloud or on-prem solution that enables the processing of machine data – logs, instrumentation data, application data, metrics, etc. – in real time, and delivers them to platform or platforms of choice.

CrowdStream: An OEM of Cribl Stream that limits the destinations that data can be delivered to CrowdStrike Falcon LogScale, CrowdStrike Falcon Next-Gen SIEM and AWS S3 buckets. It also does not support Cribl Edge as a data collection source.

Platforms relationship:

Stream acts as a data collection platform that can gather, configure and provide data to Next-Gen SIEM. Stream can collect data by either receiving it from a source or by leveraging different source APIs and pulling the data in. A high-level overview of the data flow looks like this:



Next-Gen SIEM connectors used with Stream overview:

There are currently 3 different types of Next-Gen SIEM connectors that can be leveraged with Stream: native push connectors, the generic http event collector (HEC) and the Cribl connector.

IMPORTANT NOTE:

While data can be modified within Stream by using pipelines, extreme care should be taken when doing this. Modification of the data can result in the CrowdStrike provided Next-Gen SIEM parser not being able to properly parse the data. Please validate that any modifications do not impact the CrowdStrike parser's ability to properly handle the data.

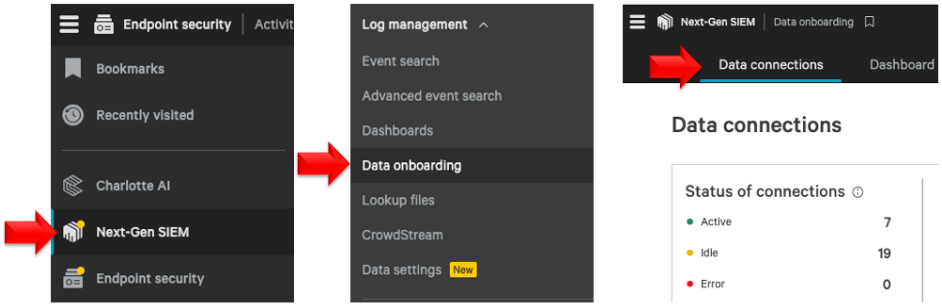
Native push connectors

Push connectors let other systems send, or “push,” data to Falcon Next-Gen SIEM. The data source drives the transfer operation and pushes new data to a CrowdStrike-provided URL whenever it’s available.

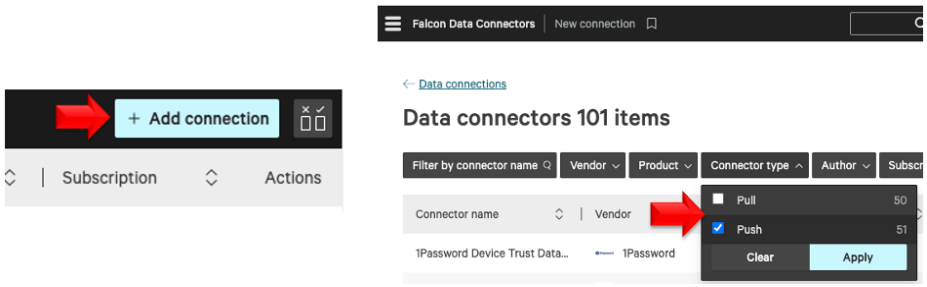
Native push connectors do have one unique feature that only applies to them. Data that’s sent to and parsed by a native collector is stored in a dedicated repository that is typically named for the vendor the data is created by. This is important to remember as things like correlation rules and other features and functions can be configured to leverage a specific repository.

Finding native push connectors

The native data connectors can be found in the Falcon UI by first navigating to Data connections dashboard in Next-Gen SIEM:



Then selecting Add connection in the bottom right corner and filtering the connectors by Connector Type:



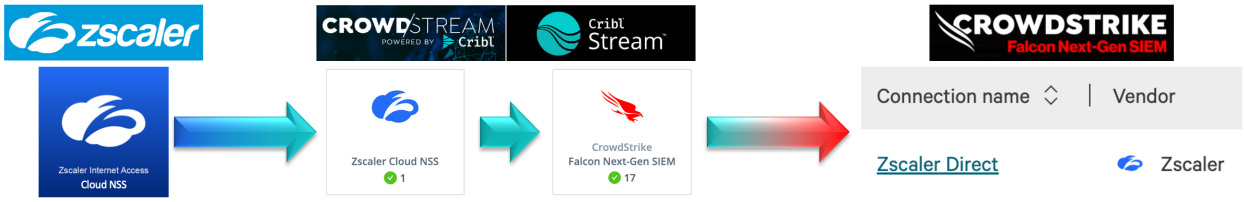
Using native push connectors with Stream

Native push connectors can be leveraged by having Stream act as a data proxy. Instead of having the data source send or "push" the data to the native Next-Gen SIEM connector, it instead sends it to Stream. The data can pass through Stream or can be processed through different pipelines before being then sent to into Next-Gen SIEM.

WARNING: **Modification of the data can result in the CrowdStrike provided Next-Gen SIEM parser not being able to properly parse the data.**

The following are examples of using Stream for data collection into Next-Gen SIEM's native push connectors:

Example 1: Zscaler Cloud NSS



In this example Zscaler's ZIA Cloud Nanolog Stream Service (NSS) is configured to send data to the Zscaler Cloud NSS source in Stream. The Zscaler Cloud NSS source is configured to pass the data through Stream to the CrowdStrike Next-Gen SIEM destination.

Generic HTTP Event Collector (HEC) connector

The Generic HTTP Event Collector (HEC) connector enables customers to push data from any source using the HTTP or HTTPS protocol for ingestion into Next-Gen SIEM. This connector is one of two that can be leveraged with Stream to provide an almost unlimited capability to pass data into Next-Gen SIEM. Any data that Stream can collect and process can then be sent to Next-Gen SIEM for parsing and ingestion.

WARNING: **Modification of the data can result in the CrowdStrike provided Next-Gen SIEM parser not being able to properly parse the data.**

The following are examples of using Stream for data collection into Next-Gen SIEM's HEC connectors:

Example 1: Proofpoint Targeted Attack Protection (TAP)

This is an example of using Stream instead of a Next-Gen SIEM native collector to collect Proofpoint TAP data. Stream's Generic REST collector has been configured to periodically poll the Proofpoint TAP SIEM REST API to collect data. In order for it to properly be processed by Next-Gen SIEM it will need to be processed through stream and broken into individual events, meaning at least an event breaker will need to be used. Depending on the filters and the desired outcome, used it may also be necessary to filter the different arrays that are returned.

Example 2: Zscaler Cloud NSS

This example is the same configuration as the example for the native Next-Gen SIEM connector, only the instead of sending the data to the native Next-Gen SIEM connector the data is pushed to Stream. This is common workflow is common for customers that are interested in ingesting the data once but being able to provide it to multiple platforms as well as customers that are looking to observe and potentially modify or tag data prior to it going into Next-Gen SIEM.

Cribl connector

The Cribl connector is a customized type of Generic HTTP Event Collector (HEC) connector and also enables customers to push data from any source using the HTTP or HTTPS protocol for ingestion into Next-Gen SIEM. This connector is one of two that can be leveraged with Stream to provide an almost unlimited capability to pass data into Next-Gen SIEM. The Cribl connector includes the ability to include the vendor's name and product (if listed) in the configuration information.

WARNING: **Modification of the data can result in the CrowdStrike provided Next-Gen SIEM parser not being able to properly parse the data.**

The following are examples of using Stream for data collection into Next-Gen SIEM's HEC connectors:

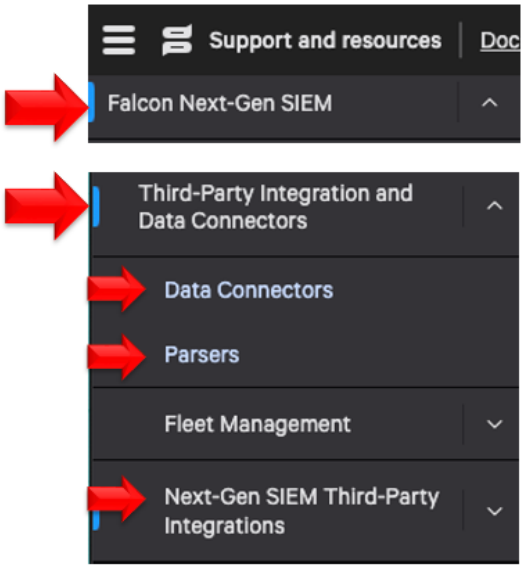
Example 1: Gigamon

In this example Gigamon is pushing data to a Raw HTTP Source in Stream instead of pushing to the Next-Gen SIEM native collector. Stream processes the data through a pipeline. The pipeline in this example does 2 actions to the data:

1. A breaker function breaks out the events from the array, making them stand alone events.
2. A mask function redacts an authentication field value that's present in the HTTP headers.

Next-Gen SIEM configurations

In addition to the information provided here it is strongly recommended that the published documentation available in the Falcon UI related to these topics are reviewed. A majority of this documentation can be found in the following location:



Retention and Timestamps

When data is sent to Next-Gen SIEM, the parser is used to identify the timestamp for the specific event. (If the parser is not able to do this then the ingested time will be used.) This is important to take into account because the data retention period for Next-Gen SIEM is established in the Next-Gen SIEM subscription. So, if the data retention period is 6 months, then the data will be retained for six months. It is critical to understand that the retention period is tied to the event's timestamp and is not based on the time the event was ingested. In other words, if an event was received with a timestamp value of 7 months ago, that data would be parsed, processed but the dropped because it would be outside the retention timeframe. However, it would still count against the ingest volume because it was properly processed and parsed into the platform. In the event that it's necessary to ingest data outside of the retention period, a new timestamp field could be created in Stream (with a value within the retention period) and the parser updated to leverage that field.

Parsers

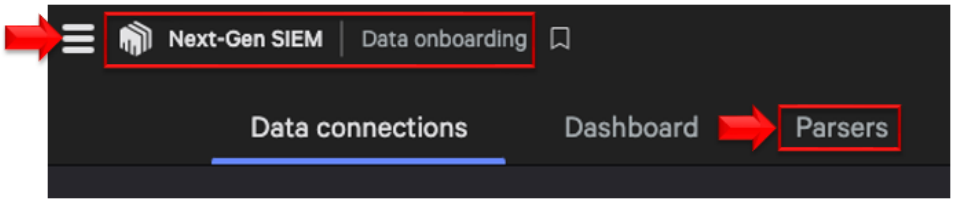
All Next-Gen SIEM connectors leverage parsers. Parsers are written in CrowdStrike Query Language (CQL) and are used to transform incoming data into searchable events that trigger detections in Next-Gen SIEM.

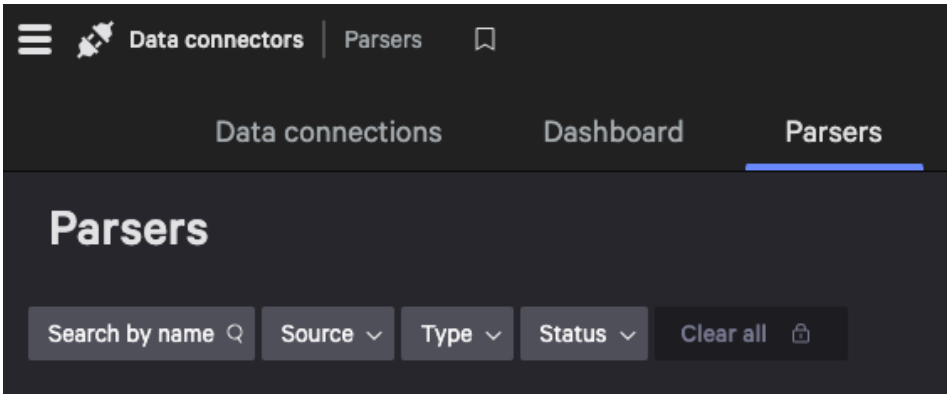
It is always best to leverage CrowdStrike or vendor authored parser to process data because they are vetted by CrowdStrike engineering. However, in some cases it's necessary to create a custom parser to handle the data. The most effective way to create custom parsers are:

1. Identify a similar CrowdStrike or vendor authored parser, clone that parser and modify it as necessary.
2. Identify if there is another customer's parser is available that can either properly parse the data or can be modified to properly parse the data. If the parser is outside the Falcon instance, import a copy of the parser and modify as required.
3. Create a custom parser from scratch, ensuring that the custom parser meets the basic parser requirements in the published documentation.

Location in the Falcon UI

Parsers can be located by navigating to the Next-Gen SIEM – Data onboarding page and selecting the Parsers tab.



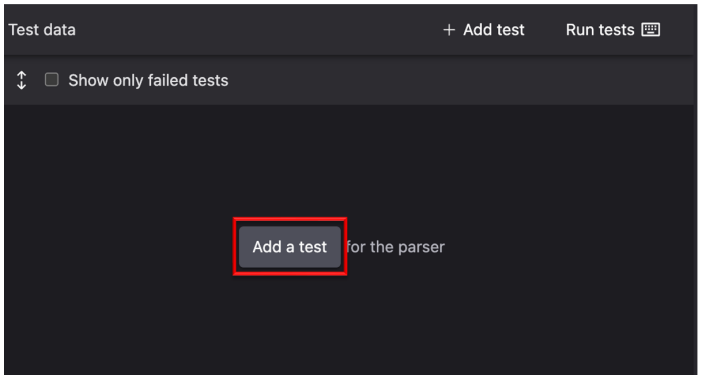


Testing Parsers

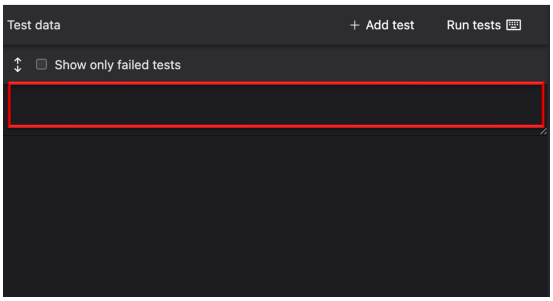
Parsers can be tested within the Falcon UI by the following process.

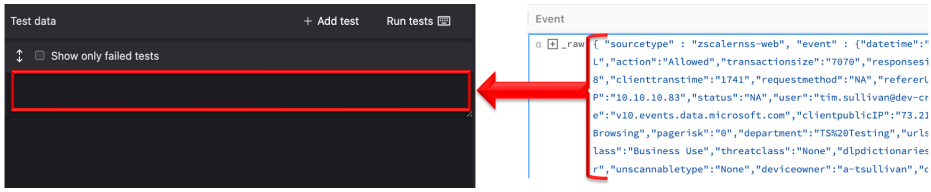
1. The most effective way to collect a sample from Stream is to use the capture feature to collect a sample of the final _raw field value.

2. Determine if there the parser to be tested is a CrowdStrike provided parser or a custom created parser.
3. CrowdStrike provided parsers do not currently allow for adding a test event. Therefore, the CrowdStrike provided parser should be cloned so that it a test event can be evaluated.
4. Open the parser that will be used to run the test and select ‘Add Test’ in the “Test data” window.

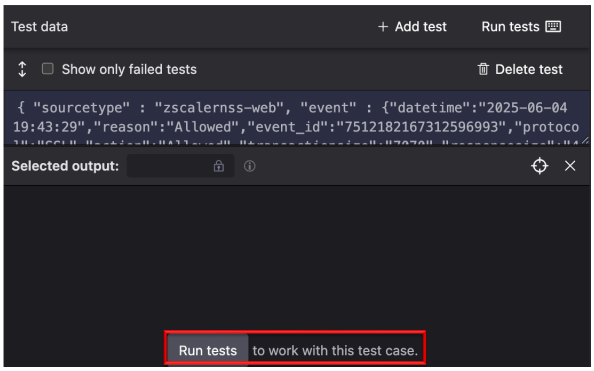


5. Select the text area that appears in the “Test data’ window and paste in the _raw data.

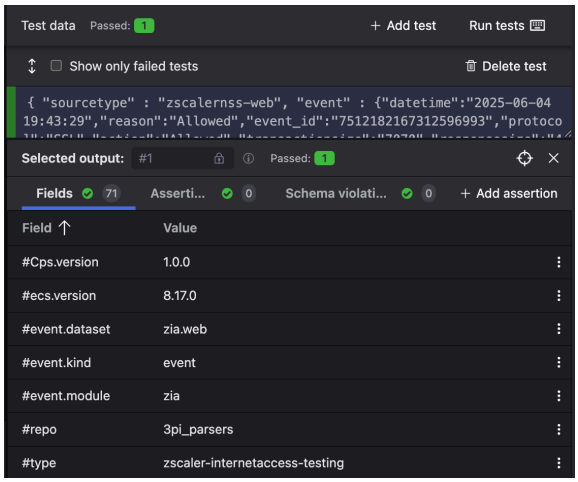




6. Select the “Run Test” button to test the event data.



7. The result should be that the test has passed, if not then validate that the _raw data is the only data that is present. If the _raw data is the proper format and syntax, it may be necessary to modify the parser. If this parser is one that's been created and is maintained by CrowdStrike, open a support ticket with CrowdStrike containing all the information and a sample of the log that is failing.



Native vendor specific connectors

Native data connectors are created by and maintained CrowdStrike and third-party vendors. There are currently two different types of these connectors:

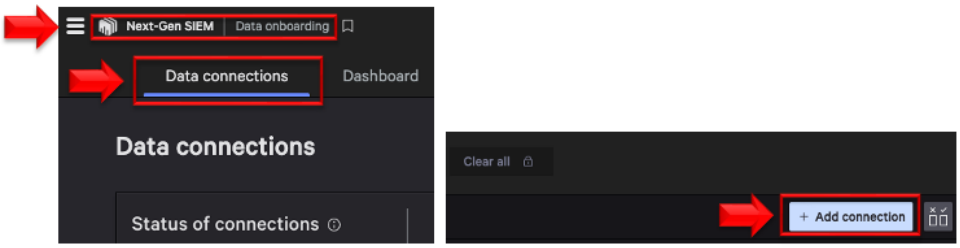
1. **Pull Connectors**: Pull Connectors contain a Data source configuration model with provider-specific fields that define how Falcon should access and retrieve data from the source. These types of connectors are not leveraged in conjunction with Stream; however, the associated parsers can often still be leveraged.
2. **Push Connectors**: Push Connectors are assigned a unique ingest URL and key by Falcon and the data source 'push' data to those URLs.

Repository Type

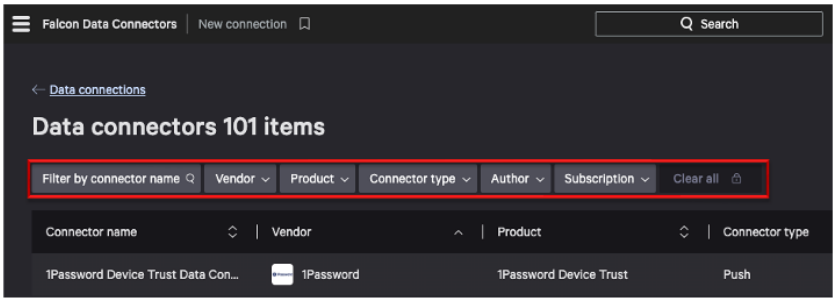
Native connectors leverage statically named repositories that are often named after the data source vendor. These static repositories are significant because they are often what is leveraged by out of the box feature and functionality with Next-Gen SIEM, such as correlation rules. **Note**: Native vendor specific connectors will always use the statically named repository for every instance of that connector type. e.g. If there are six Zscaler connectors deployed they will all push data to the 'zscaler' repository.

Location in the Falcon UI

Native connectors can be found by navigating to the Next-Gen SIEM – Data onboarding page, selecting the Data connection tab and selecting the Add connection button in the bottom right corner.

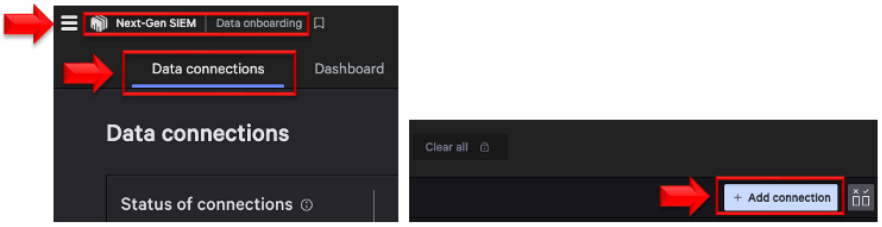


The data connectors page offered multiple filter options to use in searches. For the connectors that can be leveraged with Stream, filter on Connector type = Push.

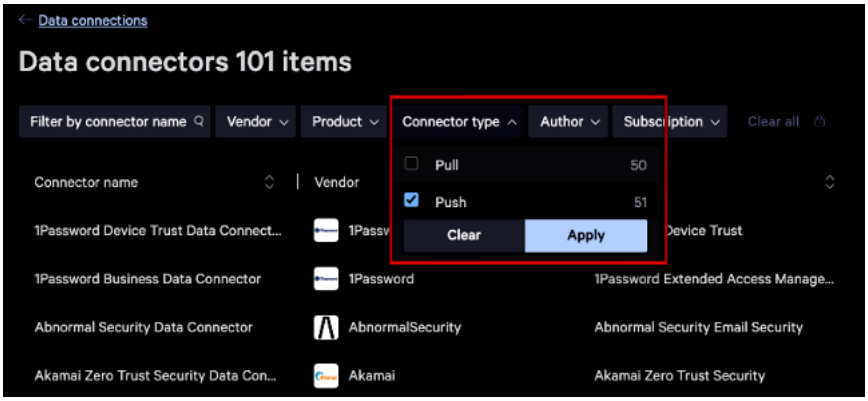


Creating a native vendor connector

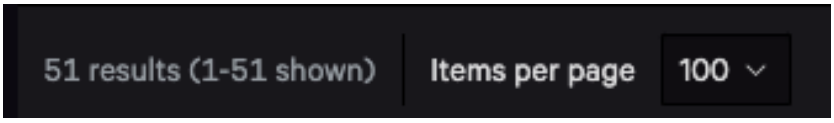
1. Native to the Next-Gen SIEM – Data onboarding page and select ‘Add connection’ in the bottom right corner.



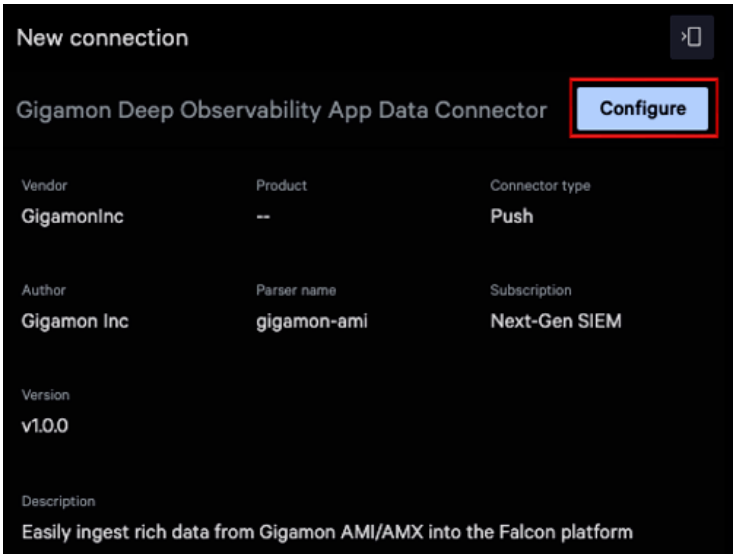
2. The only native connectors that can be leveraged with Stream are push connectors so the most effective way to locate those types of connectors is to filter on the ‘Connector type’. Select the ‘Push’ value and select ‘Apply’.



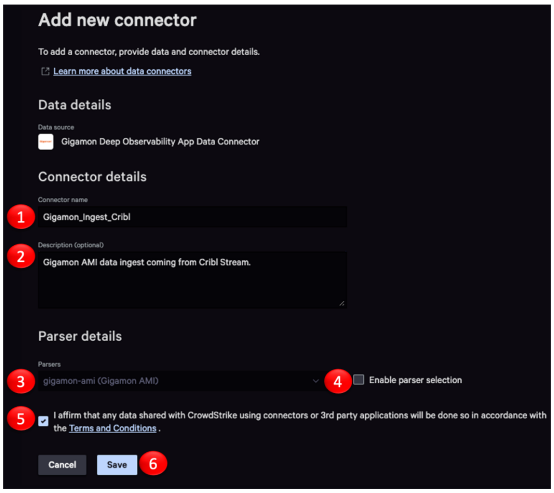
3. In order to show all the results, it's necessary to increase the number of items per page. Locate this setting at the bottom of the page and increase it to 100. Also keep in mind that the default order is alphabetical by the name in the Vendor column.



4. Locate the desired connector by either scrolling through the list or using additional filters to narrow down the list. Select the connector in the list and then select 'Configure' in the right-hand pane.

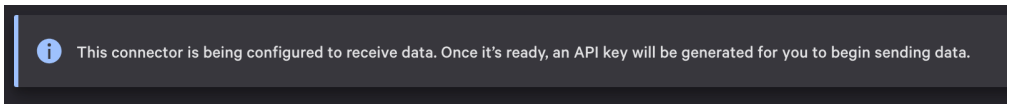


5. Configure and save the Connector.

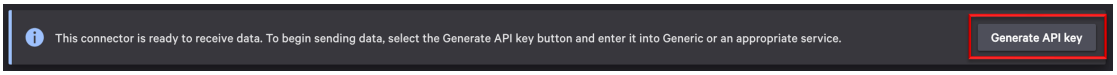


1. **Connector Name**: A unique name which will be displayed in the UI page. Including 'Stream' or 'Cribl' in the name can help identify native connectors associated with Stream.
2. **Connector Description**: A description that will be viewable in the connector details page.
3. **Parser Selection**: The parser that's currently assigned to the connector. It is highly recommended to ensure that the most up to date CrowdStrike provided parser is selected.
4. **Enable parser selection**: (**Optional**) This option will enable a different parser to be selected for the connector. This should only be used if the current parser has been decommissioned and a new CrowdStrike version needs to be selected. Custom parsers should only be used when absolutely necessary.
5. **Terms & Conditions**: Indicates the review and acceptance of the terms and conditions.
6. **Save**: Saves the connector configurations

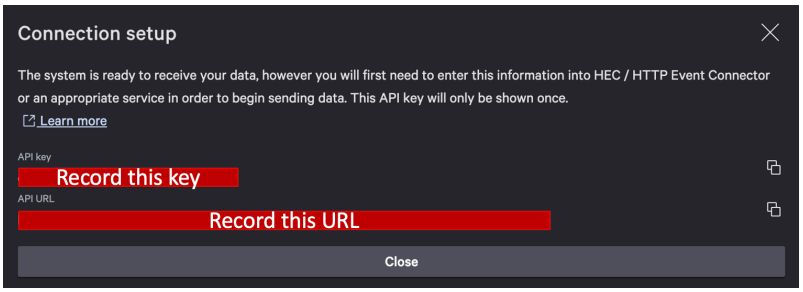
6. Once saved the connector an information window will appear stating that the connector is being configured, refresh the page.



7. Select 'Generate API key' to create an API URL and API token. The API token will only be available at this point, if the API token is lost then it will need to be regenerated (which will invalidate the current token).



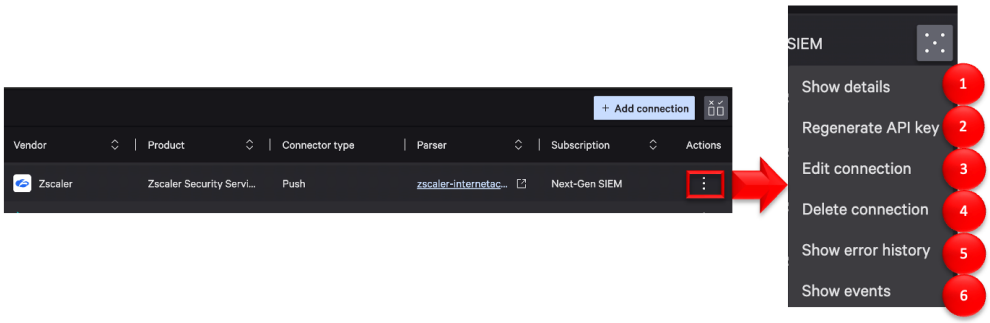
8. Record the API URL and API token. *NOTE: The API token is only available during this time. If the API token becomes unknown, it will need to be regenerated and the current one will become invalid.



9. Select 'Close' to close the window. The connector is now configured and will be in a 'Pending' state until it receives data.

Working with an Existing Native Connector

If a connector has been created and is visible on the connector page, the 'Actions' menu can be used to interact with it. Clicking on the 'Actions' menu (represented by 3 vertical dots) to the far right of the connector will expand the action menu.



1. **Show details**: Opens the connector details page
2. **Regenerate API Key**: Opens a popup window to generate a new API key, invalidating the previous key
3. **Edit Connection**: Opens the connector configuration page
4. **Delete Connection**: Opens a popup window to delete the connector
5. **Show error history**: Opens an advanced search page, populated with a search for connector errors for the past 1 hour
6. **Show event**: Opens an advanced search page, populated with a search for events for the connector for the past hour

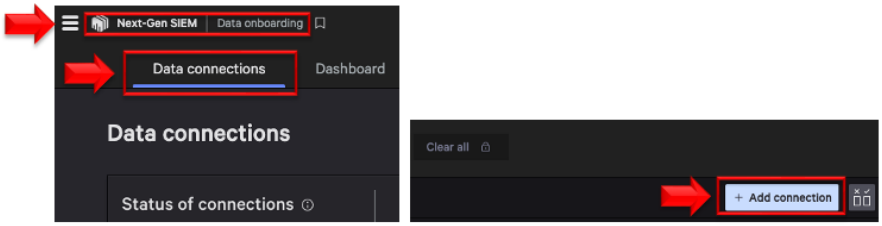
HEC connector

The HTTP Event Connector (HEC) is a generic native connector that enables ingesting data from any source using the HTTP or HTTPS protocol.

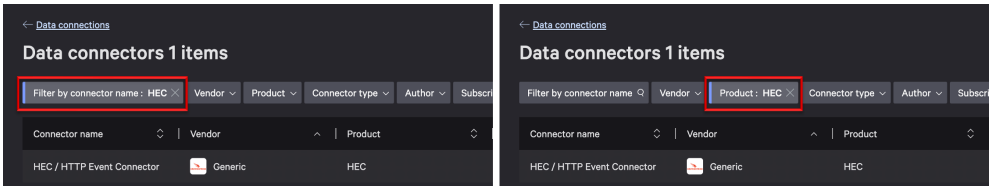
Repository Type: A dynamic repository is created (typically starting with the prefix '3pi_auto_raptor_') for each connector that is configurated. This may require some of the out of the box features and functionality to be updated to properly be able to locate the necessary data.

Creating an HEC connector

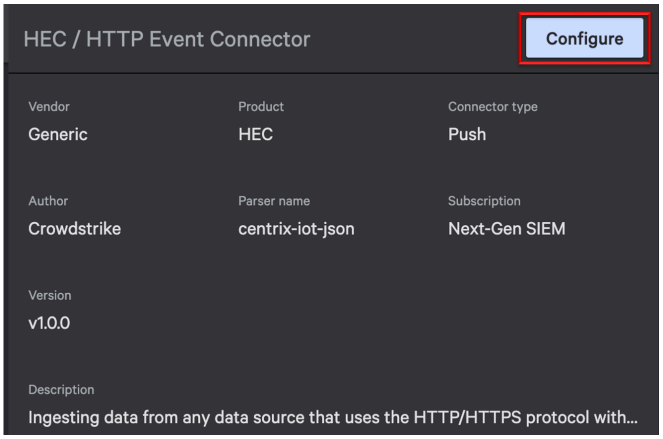
1. Native to the Next-Gen SIEM – Data onboarding page and select 'Add connection' in the bottom right corner.



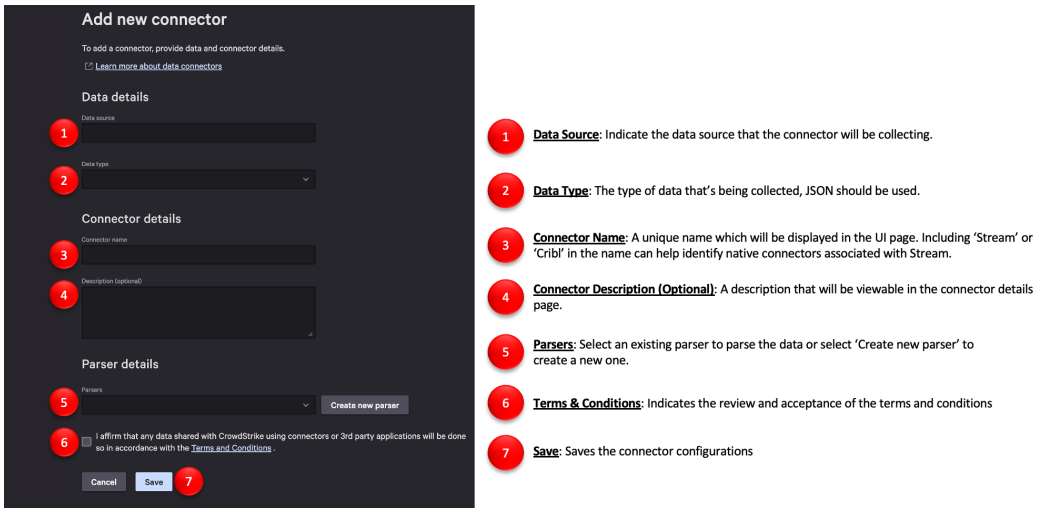
2. Use the Search feature or filter by Product 'HEC' to locate the HEC/HTTP Event Connector.



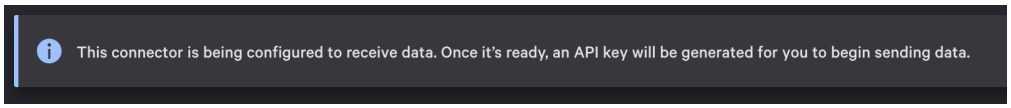
3. Select the HEC and then select 'Configure' in the top right corner.



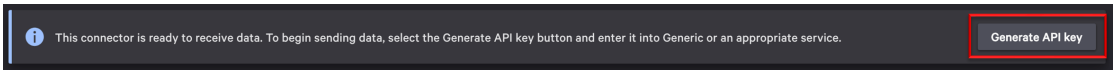
4. Configure and save the HEC connector.



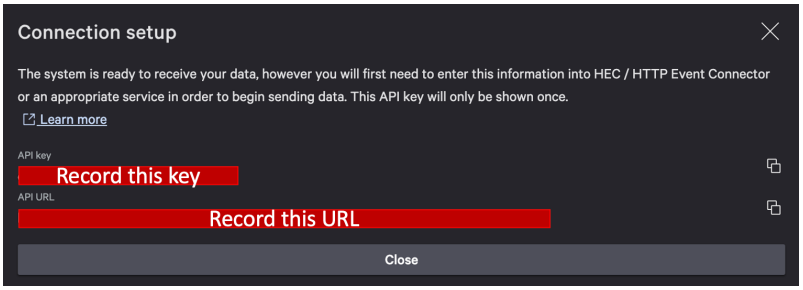
5. Once saved the connector an information window will appear stating that the connector is being configured, refresh the page.



6. Select 'Generate API key' to create an API URL and API token. The API token will only be available at this point, if the API token is lost then it will need to be regenerated (which will invalidate the current token).



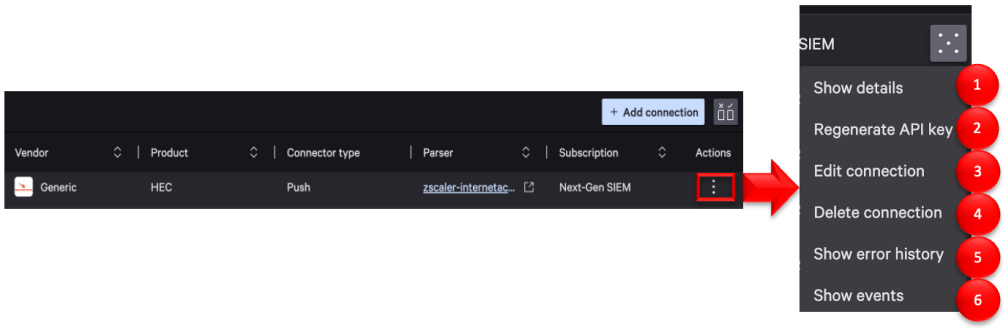
7. Record the API URL and API token. *NOTE: The API token is only available during this time. If the API token becomes unknown, it will need to be regenerated and the current one will become invalid.



8. Select ‘Close’ to close the window. The connector is now configured and will be in a ‘Pending’ state until it receives data. To see recent events that have been collected by the connector, select the ‘Actions’ menu and select ‘Show events’ (see Working with an Existing HEC Connector)

Working with an Existing HEC Connector:

If a connector has been created and is visible on the connector page, the ‘Actions’ menu can be used to interact with it. Clicking on the ‘Actions’ menu (represented by 3 vertical dots) to the far right of the connector will expand the action menu.



1. **Show details**: Opens the connector details page
2. **Regenerate API Key**: Opens a popup window to generate a new API key, invalidating the previous key
3. **Edit Connection**: Opens the connector configuration page
4. **Delete Connection**: Opens a popup window to delete the connector
5. **Show error history**: Opens an advanced search page, populated with a search for connector errors for the past 1 hour
6. **Show event**: Opens an advanced search page, populated with a search for events for the connector for the past hour

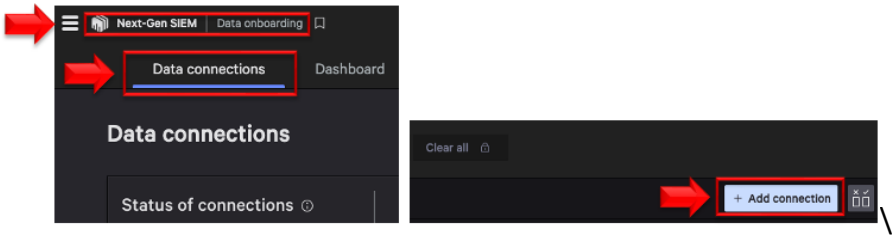
Cribl connector

The Cribl connector is a customized version of the HTTP Event Connector (HEC) and is designed to ingesting data from Stream using the CrowdStrike Next-Gen SIEM Stream destination.

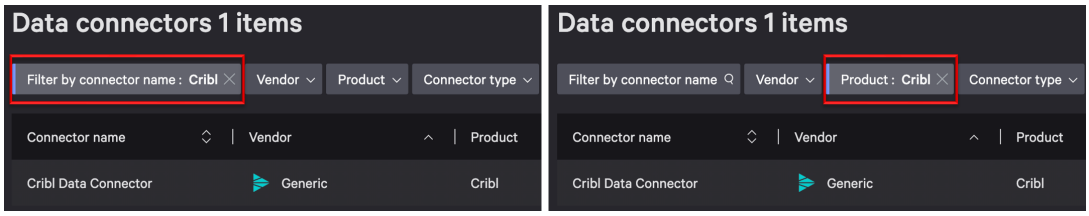
Repository Type: A dynamic repository is created (typically starting with the prefix '3pi_auto_raptor_') for each connector that is configurated. This may require some of the out of the box features and functionality to be updated to properly be able to locate the necessary data.

Creating a Cribl connector

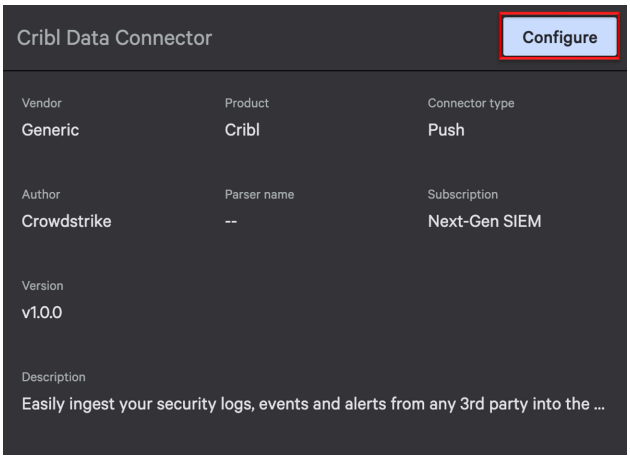
1. Native to the Next-Gen SIEM – Data onboarding page and select 'Add connection' in the bottom right corner.



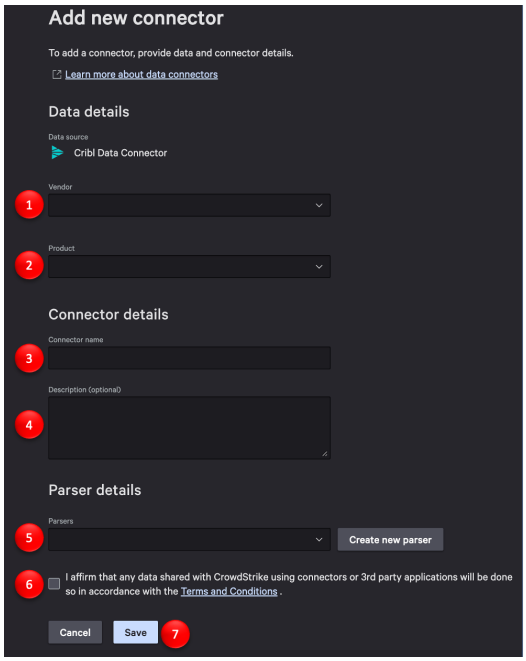
2. Use the Search feature or filter by Product 'Cribl' to locate the Cribl Connector.



3. Select the Cribl connector and then select ‘Configure’ in the top right corner.

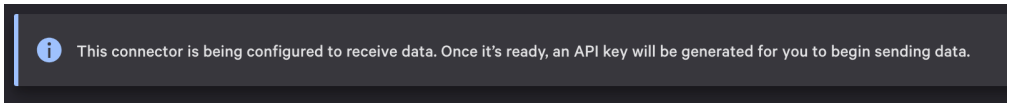


4. Configure the Cribl connector and save.

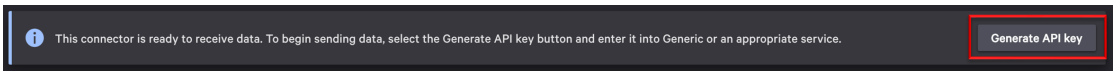


1. **Vendor**: Select the vendor that the data is coming from or select ‘Generic’
2. **Product**: Select the vendor product that the data is coming from or select ‘Generic’.
3. **Connector Name**: A unique name which will be displayed in the UI page.
4. **Connector Description (Optional)**: A description that will be viewable in the connector details page.
5. **Parsers**: Select an existing parser to parse the data or select ‘Create new parser’ to create a new one.
6. **Terms & Conditions**: Indicates the review and acceptance of the terms and conditions
7. **Save**: Saves the connector configurations

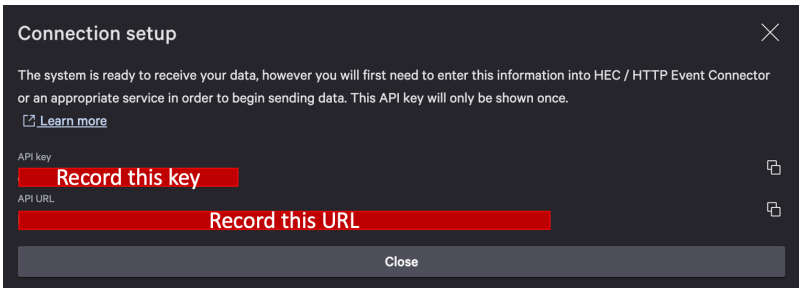
5. Once saved the connector an information window will appear stating that the connector is being configured, refresh the page.



6. Select ‘Generate API key’ to create an API URL and API token. The API token will only be available at this point, if the API token is lost then it will need to be regenerated (which will invalidate the current token).



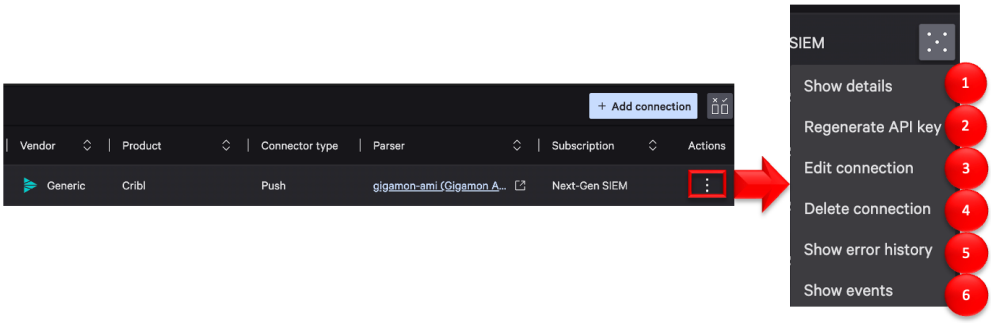
7. Record the API URL and API token. *NOTE: The API token is only available during this time. If the API token becomes unknown, it will need to be regenerated and the current one will become invalid.



8. Select ‘Close’ to close the window. The connector is now configured and will be in a ‘Pending’ state until it receives data.

Working with an Existing Cribl Connector:

If a connector has been created and is visible on the connector page, the 'Actions' menu can be used to interact with it. Clicking on the 'Actions' menu (represented by 3 vertical dots) to the far right of the connector will expand the action menu.



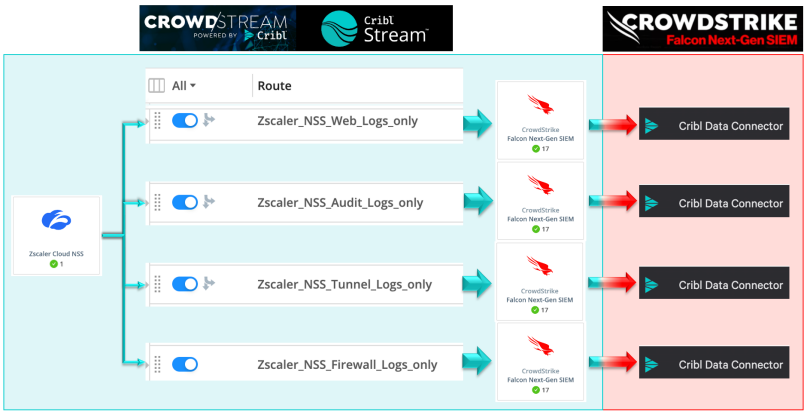
1. **Show details**: Opens the connector details page
2. **Regenerate API Key**: Opens a popup window to generate a new API key, invalidating the previous key
3. **Edit Connection**: Opens the connector configuration page
4. **Delete Connection**: Opens a popup window to delete the connector
5. **Show error history**: Opens an advanced search page, populated with a search for connector errors for the past 1 hour
6. **Show event**: Opens an advanced search page, populated with a search for events for the connector for the past hour

Specific Considerations for HEC and Cribl Connectors

The HEC and Cribl connectors do not currently have an EPS limit, which means they're designed to handle as many events that can be sent to it. However, there are some considerations that needs to be taken into account when using these connectors.

1. **Use at least one connector per event source**. For every event source within Stream, there should be at least one Next-Gen SIEM destination configuration and one Next-Gen SIEM Connector dedicated to that destination configuration. There are two specific reasons that specific event sources need to be assigned to dedicated HEC or Cribl connectors. The first is that the parser that's assigned to that connector is going to map specific fields that are used within Next-Gen SIEM as well as provide specific data tagging to every event. The second is that those connectors are going to create a repo to store the collected data.

2. **Using multiple connectors for high volume event sources**: While it's highly recommended to use at least one connector per event sources, that does not mean that only one connector needs to be used. For sources that have a high volume of events or event types where it might be more beneficial to segment them out accordingly, multiple destinations in Stream can be created and mapped to multiple destinations within Next-Gen SIEM. This kind of configuration allows more events to be processed at a time, while also providing the ability to have more granular search options (e.g. searching by a specific repository).
   Implementing this solution in Stream would be best created by leveraging routes with specific filtering statements with only the last route being marked as 'final'. For example, filtering for just 4 different event types from Zscaler's Cloud NSS platform might look like this (Cribl connectors are all leveraging the same ZIA parser):



The Next-Gen SIEM endpoint URLs used with Stream

The /services/collector endpoint

The /services/collector endpoint in Next-Gen SIEM is the default endpoint for the Generic HEC and Cribl connectors. It's designed to accept a JSON formatted message and processes specific fields names within that data.

NOTE: While the message format is JSON the event format does not have to be. For example, PaloAlto Firewall logs will be sent as a JSON message but the event itself will be CSV.

While there is a /services/collector endpoint in LogScale, it's functionality is different than the one used in Next-Gen SIEM. This is important to keep in mind when reviewing the available documentation because while the Next-Gen SIEM endpoint will accept all the same fields, it will not actually collect or parse them all.

The following matrix represents the standard HEC fields and how/if they're leveraged in Next-Gen SIEM.

Field Name	Collected by Endpoint	Processed by Parser	Description
time	no	N/A	Time in seconds since January 1, 1970 in UTC.
timezone	no	N/A	Used to describe the time zone in which the event happened.
index	no	N/A	Optional name of the repository to ingest into
sourcetype	no	N/A	Translated to the #type tag on ingestion
source	no	N/A	Translated to the @source metadata field on ingestion
host	no	N/A	Translated to the @host field on ingestion
event	YES	YES	This is the field that contains the event data that will be displayed in the @rawstring field
fields	YES	no	These can be additional fields that can be added to the event but will not be parsed or part of the @rawstring field Use this field for tagging.

Stream data formatting and Next-Gen SIEM

Stream can populate the 'event' and 'fields' fields but there are some specific syntax considerations that need to be taken into account.

Populating the 'event' field

The 'event' field is the only field that Next-Gen SIEM will accept and parse as the event from the data source. It is also the only data that will be recorded in the '@rawstring' field within the Next-Gen SIEM event. The only recommended way to populate this field in Stream is to have the data contained within an '_raw' field.

Considerations and Warnings:

- Sending a message to Next-Gen SIEM from Stream without the '_raw' field will result in the entire message (with the exception of some Stream system fields and reserved fields) being handled as both the 'event' and 'fields' field values.
- Next-Gen SIEM does not currently support sending JSON arrays with multiple events and will process a '_raw' field containing a JSON array as single JSON object.
- Attempting to rename the '_raw' field to 'event' will result in the entire message (with the exception of some Stream system fields and reserved fields) being handled as both the 'event' and 'fields' field values and can also impact correct parsing.
- Attempting to manually create an 'event' field (while retaining '_raw') will result in the 'event' field being included in the 'fields' field values.

Populating the 'fields' field

The 'fields' field is populated automatically by Stream and cannot be done manually. All fields that are not within the '_raw' field will be added to the 'fields' field (including Stream system fields) and will be included in the Next-Gen SIEM event but will not be parsed nor will they be included in the '@rawstring' field value.

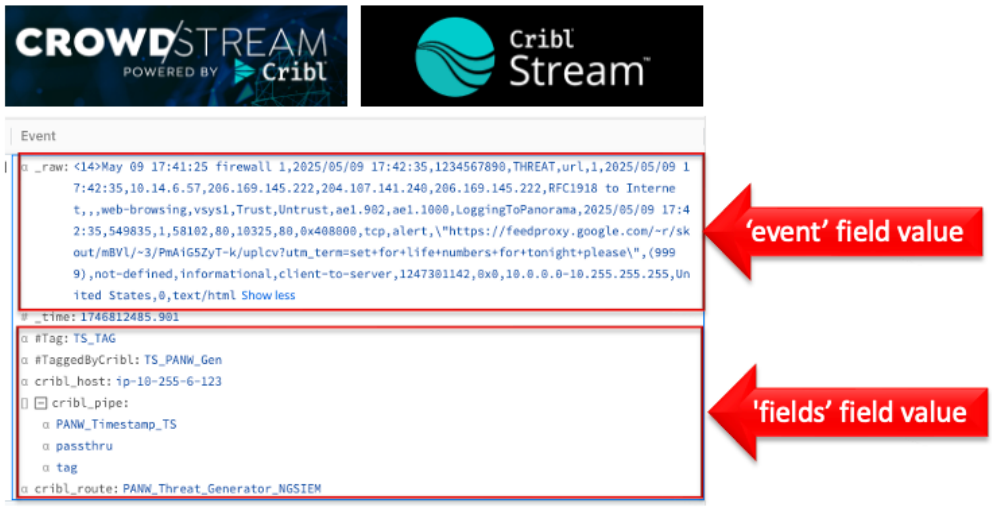
Considerations and Warnings:

- Sending a message to Next-Gen SIEM from Stream without the '_raw' field will result in the entire message (with the exception of some Stream system fields and reserved fields) being handled as both the 'event' and 'fields' field values.
- Attempting to manually create a 'fields' field will result in a field within 'fields' with the name "field".
- The field values for 'fields' are only processed at the first level, meaning that if a 'fields' field value is a JSON array or multiple JSON objects, they will not be broken out. The JSON array or JSON objects will be a single field value.

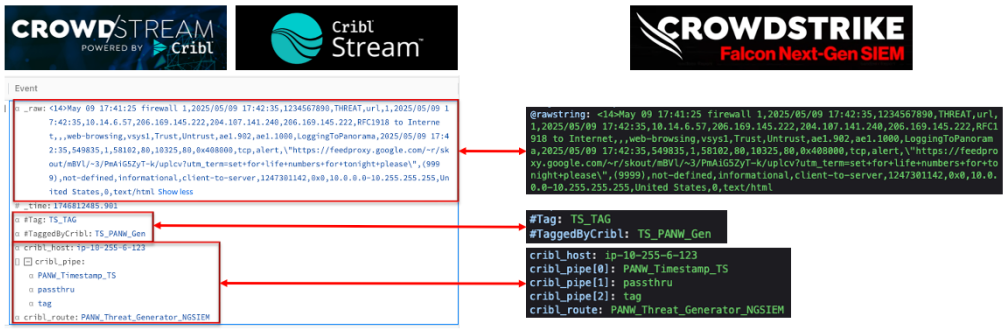
Proper Stream event syntax example:

The following is an example of a Palo Alto firewall event that's being sent from Stream to Next-Gen SIEM. The event contains an '_raw' field as well as additional fields, one of which is a Next-Gen SIEM tag, and was sent using Stream's Next-Gen SIEM destination.

The event capture before the destination:



- The _raw data is going to be put into the 'event' field. This data is what Next-Gen SIEM will parse and represent in the @rawstring field.
- The other fields will be put into the 'fields' field. These will also be present in the Next-Gen SIEM events and searchable; however, they will not be parsed upon ingestion. Any fields beginning with the hashtag or pound symbol (#) will create Tags in Next-Gen SIEM.



Improper Stream event syntax examples:

1. Attempting to manually create an ‘events’ field based off the ‘_raw’ field.

2. Renaming the ‘_raw’ field to another name, like ‘event’.

3. Attempting to manually add a ‘fields’ field / passing multiple JSON objects in a ‘fields’ field.

Including addition data and Next-Gen SIEM Tags in Stream

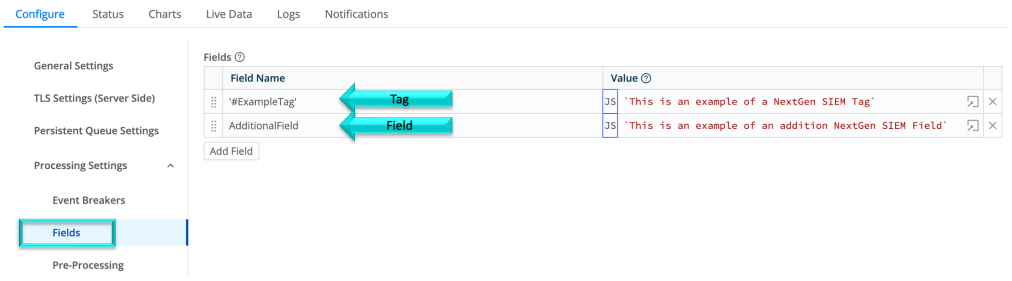
In some cases, there may be a need or desire to include additional information or Next-Gen SIEM tags into data begin sent to Next-Gen SIEM. This can be accomplished using the /services/collector endpoint and including the additional fields and tags outside of the event field which in Stream mean including it outside the _raw field.

Important to Keep in Mind:

- In order for the data to be processed correctly there needs to be an '_raw' field in the Stream event. This field should contain the event data that's been collected from the source and will be the event data that makes up the Next-Gen SIEM event.
- Next-Gen SIEM tags are fields that begin with a hashtag or pound symbol (#). In order to properly create this field name in Stream it's necessary to wrap the name in single quotes.
- Additional fields and tags are added outside of the '_raw' field and are not considered part of the event and will not be parsed by the Next-Gen SIEM parser.

1. The Source configuration: process settings: fields

Additional Next-Gen SIEM fields and tags can be added to a Stream source input configuration, under 'Process Settings' is the Fields configuration. This configuration allows for fields to be added to every event for that input. In the following example there are 2 fields that are going to be included in every event that gets received.

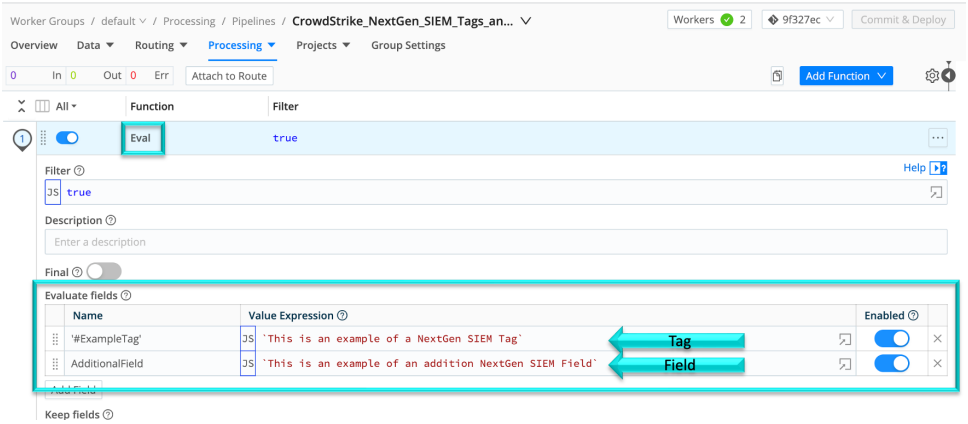


- The '#ExampleTag' is a Next-Gen SIEM tag, which is why it starts with a # symbol. Since the Next-Gen SIEM tag starts with a symbol, the syntax requires that it be wrapped in single quotes.
- The 'AdditionalField' is just a normal JSON field and does not require quotes.

NOTE: The syntax (as shown in the example above) requires that the Field Name is wrapped in single quotes and the Value, because it's a JavaScript template string, is wrapped in backticks.

2. **Processing and Post-Processing Pipeline**

Additional Next-Gen SIEM fields and tags can be added to a Stream pipeline configuration using an Eval function. This will add the fields and the tags to the events that are processed by the pipeline.

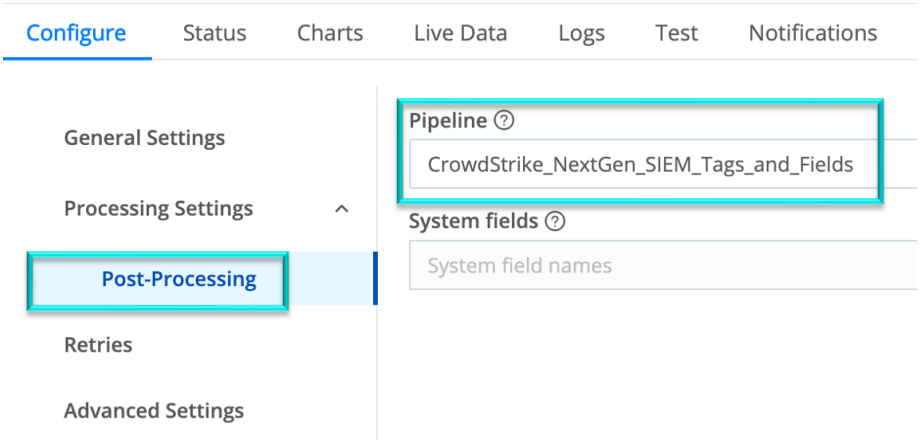


- The '#ExampleTag' is a Next-Gen SIEM tag, which is why it starts with a # symbol. Since the Next-Gen SIEM tag starts with a symbol, the syntax requires that it be wrapped in single quotes.
- The 'AdditionalField' is just a normal JSON field and does not require quotes.

NOTE: The syntax (as shown in the example above) requires that the Name field is wrapped in single quotes and the Value Expression, because it's a JavaScript template string, is wrapped in backticks.

This function can be applied as either a processing or post-processing pipeline. **It is not recommended** to use in a pre-processing pipeline, to apply tags and additional fields in the source configuration use the Processing Settings: Fields configuration above.

The post-processing pipeline if configured in the destination configuration:



The /services/collector/raw Endpoint

The /services/collector/raw endpoint in Next-Gen SIEM is only designed to process the data passed in the 'event' field and has to be passed in string format. This endpoint is able to process multiple events in a single post, but the events will be handled as strings and use a new line delimiter to separate multiple events. This endpoint is never considered the 'default' endpoint and will not be presented when configuring an HEC or Cribl connector in the Falcon UI. The only way in which this endpoint can be leveraged is by manually appending '/raw' to the end of the services collector endpoint.

While there is a /services/collector/raw endpoint in LogScale, its functionality is different than the one used in Next-Gen SIEM. This is important to keep in mind when reviewing the available documentation because while the Next-Gen SIEM endpoint will accept all the same fields, it will not actually collect or parse them all.

The following matrix represents the standard HEC fields and how/if they're leveraged in Next-Gen SIEM.

LogScale HEC field usage in NextGen SIEM /service/collector/raw endpoint

Field Name	Collected by Endpoint	Processed by Parser	Field Description
time	no	N/A	Time in seconds since January 1, 1970 in UTC.
timezone	no	N/A	Used to describe the time zone in which the event happened.
index	no	N/A	Optional name of the repository to ingest into
sourcetype	no	N/A	Translated to the #type tag on ingestion
source	no	N/A	Translated to the @source metadata field on ingestion
host	no	N/A	Translated to the @host field on ingestion
event	YES	YES	This is the field that contains the event data that will be displayed in the @rawstring field
fields	no	no	These can be additional fields that can be added to the event but will not be parsed or part of the @rawstring field Use this field for tagging.

Stream data formatting and Next-Gen SIEM

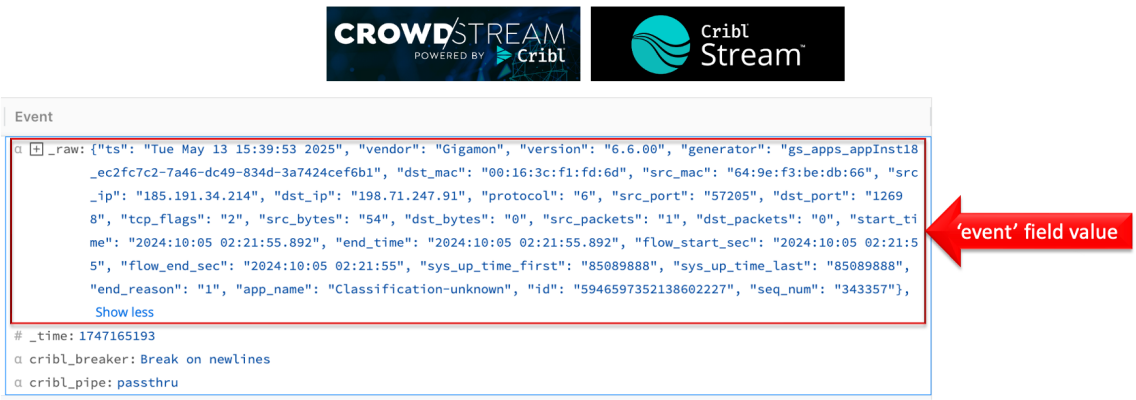
Stream can populate the ‘event’ but there are some specific syntax considerations that need to be taken into account.

Populating the ‘event’ field

The ‘event’ field is the only field that Next-Gen SIEM will accept and parse as the event from the data source. It is also the only data that will be recorded in the ‘@rawstring’ field within the Next-Gen SIEM event. The only recommended way to populate this field in Stream is to have the data contained within an ‘_raw’ field.

Proper Stream event syntax example:

The following is an example of a Gigamon event that’s being sent from Stream to Next-Gen SIEM. The event contains an ‘_raw’ field, as well as additional fields, and was sent using Stream’s Next-Gen SIEM destination.



- The _raw data is going to be put into the ‘event’ field. This data is what Next-Gen SIEM will parse and represent in the @rawstring field.
- The other fields, such as those in the ‘fields’ field, will not be processed by NGSIEM or included in the event. Only single line in the ‘event’ field will be parsed, processed and presented.

Considerations and Warnings:

- Sending a message to Next-Gen SIEM from Stream without the '_raw' field will result in the entire message (with the exception of some Stream system fields and reserved fields) being handled as the '_raw' field value.
- This endpoint does not support the 'fields' field, so it is not possible to send additional fields (outside the '_raw' field) or Next-Gen SIEM tags.
- Each line of the 'event' field will be handled as its own event.

Improper Stream event syntax examples:

1. Renaming the '_raw' field to another name, like 'event'.

Stream configurations

Unless specifically indicated, the following information applies to both CrowdStream and Cribl Stream.

Stream sources

Essentially any data source that can be configured in Stream can be used to collect data for Next-Gen SIEM. Some data sources may structure data in such a way that it can simply 'pass through' Stream and into Next-Gen SIEM, others may only be required to be separated into individual events while others may more involved modifications.

The most effective and efficient way to ensure that the data will be properly be properly parsed by CrowdStrike supported parsers is to retain the event data (the data contained in the '_raw' field value) in a structure and syntax as close to original as possible. In the event that there needs to be data modification, it's strongly recommended that the event data be tested against a copy of the data parser in Next-Gen SIEM prior to enable the sending of data.

Considerations

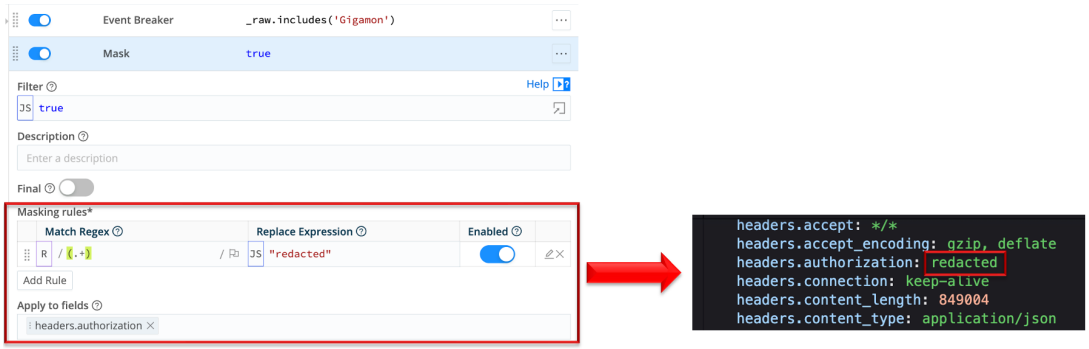
When leveraging the data sources in Stream, it is recommended that several test collections be completed with the source input to get a firm handle on the type and the structure of the data that will be collected. It is also recommended to save samples of these events to leverage in any additional testing that might need to be done.

1. **Multiple events in a single payload**: In the event that a source provides multiple events in a single payload, the best course of action is to use an event breaker (either in the source configuration or within a pipeline) to separate out the individual events prior to any modifications and/or sending them to Next-Gen SIEM. The only time this would not be necessary is if the source provided the events so that each event was on its own line and the data was being sent to a /services/collector/raw Next-Gen SIEM endpoint.

2. **Additional fields present in certain Stream Source types**: The method in which Stream is collecting the data can impact the data that is passed to Next-Gen SIEM. For example, a 'Raw HTTP' source will include a 'headers' field when the data is collected. This field is outside of the _raw field and if left in place will be sent to and processed by Next-Gen SIEM as part of the 'fields' field. At the very least, depending on the volume of events, this could increase amount of data ingested and potentially impact licensing costs. The more serious concern would be the potential for sensitive information to be inadvertently exposed.

In following example, an event that's been collected by a 'Raw HTTP' source and contains the 'header' field which has an authorization token in it. If this data is configured to pass through Stream and go directly to Next-Gen SIEM, this token will be included and visible in the event because it will be included in the 'fields' field.

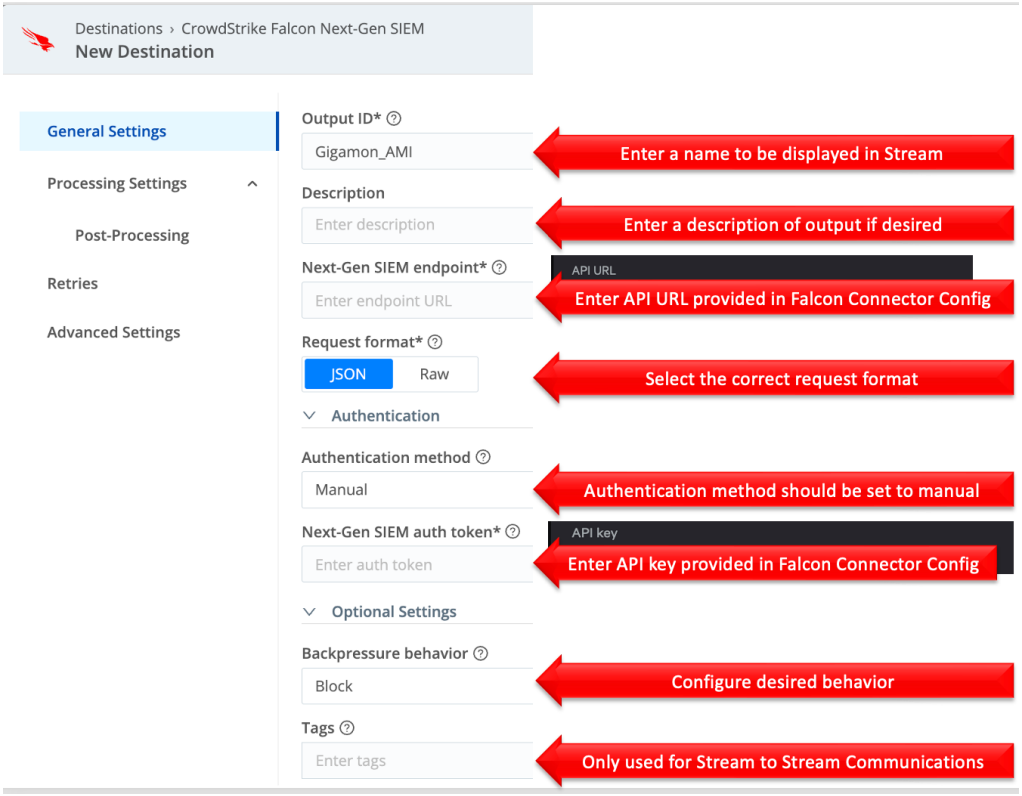
This could represent a security issue and as such either the 'headers' field would need to be removed in Stream or the field value for 'authorization' would need to be masked. The example below shows a pipeline to process Gigamon data. The first function is an event breaker function because Gigamon data is provided in a JSON array and the second is a masking function replacing the 'authorization' field value with the word 'redacted'.



Stream's Next-Gen SIEM destination

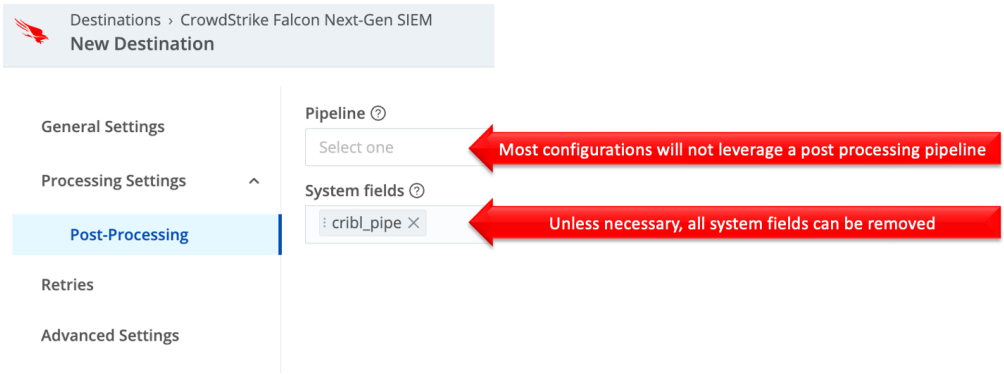
General Settings

The general settings for the Next-Gen SIEM destination will require a connector to be configured in Falcon before it can be completely configured. This is because the Next-Gen SIEM endpoint and authorization token (API key) will need to be inputted.



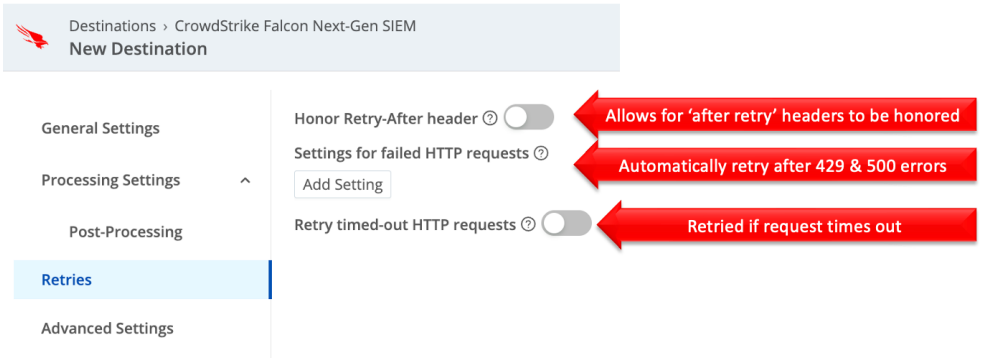
Process Settings

The process settings are not really necessary for most Next-Gen SIEM configurations. The majority of data manipulation (if any) is typically done in the source or processing pipeline vs a post processing pipeline. In addition, it's not necessary to include the Cribl system fields unless specifically desired. If the Cribl system fields are left for the purpose of identifying the data as coming from Cribl, it is recommended to leverage a Next-Gen SIEM tag for the data instead.



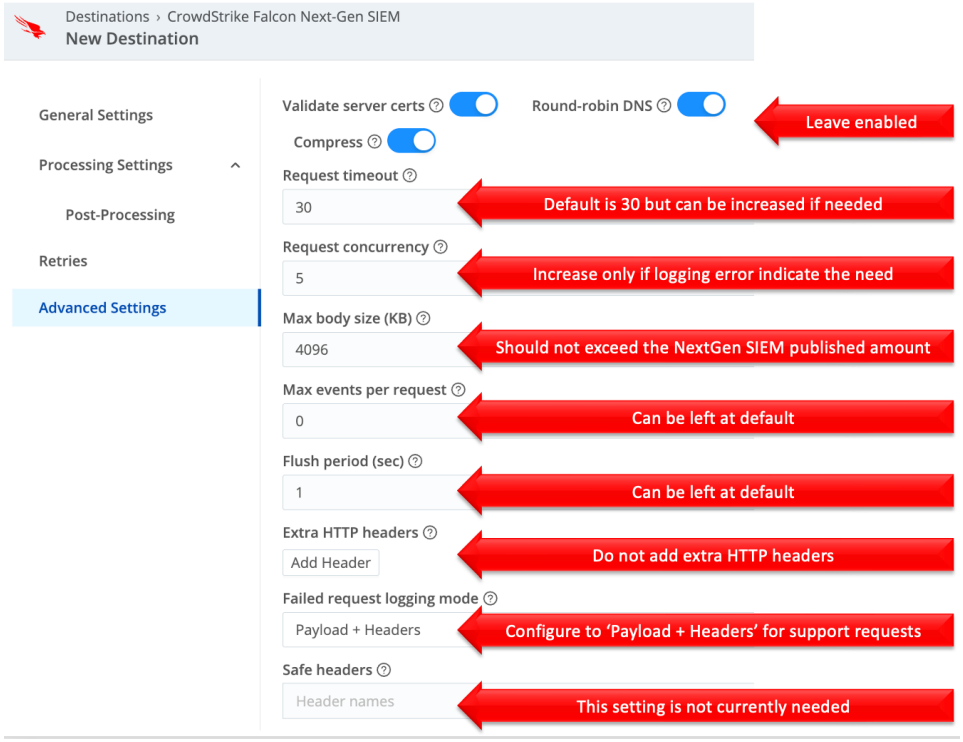
Retries

The retries settings are not configured by default. The 'Settings for failed HTTP requests' and the 'Retry timed-out HTTP requests' are two that can be configure in the event that Stream encounters retriable errors in sending data to Next-Gen SIEM.



Advanced Configuration

The advanced configuration settings are most often adjusted to remediate issue when communication with Next-Gen SIEM. This is because the settings are dependent on the data source and the size/volume of data that's being sent. In most cases the default configurations can be retained to start with, with the exception of the 'Failed request logging mode', which should be set to 'Payload + Headers' to capture as much information as possible if there is an issue. This information should be provided in any support requests to either Cribl or CrowdStrike.



Common issues and troubleshooting

Duplicate data in Next-Gen SIEM

Duplication of data can take place if certain capabilities and configurations within Stream aren't configured or tuned in correctly.

Source Configuration

The configuration of the data source is one area that can cause duplication of data, typically this is because there are timing considerations that need to be not only taken into account upon creation but may need to be tuned afterward. Some examples:

- **Custom REST Collector**: The Custom Rest Collector can be configured to run on a scheduled basis. This is typically done in conjunction with a timestamp value in the data being collected.
    - The first configuration to check is that if a timestamp field is being used, that it's the proper timestamp. Leveraging the wrong timestamp that increments outside the desired collection scheme, may result in the dataset being collected more than desired.
    - The second configuration to check is the scheduled time for the collection vs the time window of the data. For example, if the API call is constructed to collect the matching data for the last 60 minutes, then the scheduled run time should also be 60 minutes. If the schedule collection was every 30 minutes, then the 30-minute overlap could result in duplication of data.
- **AWS S3 and SQS Collectors**: The AWS S3 and SQS collectors do have some configurations that can cause duplication data to be collected.
    - **S3 Collector: Path** – If there are multiple S3 collectors targeting the same S3 bucket, the Path configuration needs to be configured appropriately in order to avoid data duplication.
    - **SQS Collector: Advanced Settings: Visibility Timeout** – The visibility timeout configuration is set to 'hide' the message while a client is collecting the data. In the event that it takes longer than the timeout value, the message will be returned to the message queue and can be collected by another client, resulting in duplication of data.

Route Configuration

Routing configurations are another area where potential configurations can cause the data to be processed multiple times resulting in duplications.

- **Data matches multiple filters**: If the data that's sent to Routes matches multiple route filters, then that data can be processed multiple times and can potentially be sent multiple times to Next-Gen SIEM. The best ways that this can be prevented is to ensure that the Route filters are as specific as possible.
- **Not flagging a Route as 'Final'**: If the data doesn't match a Route that's marked as 'Final' then it will continue to be evaluated by other routes, may match the filter applied to those routes and be processed. The way to prevent this is to Identify the last (or only) Route that will handle the data and ensure that 'Final' is configured for that Route.

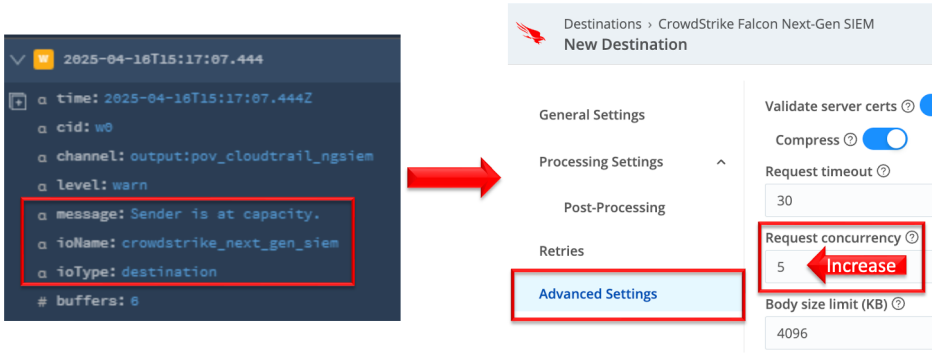
Destination Configuration

Destination configurations aren't a prone to producing duplicative events but it is prudent to review the settings/configurations of any that have the potential to cause them.

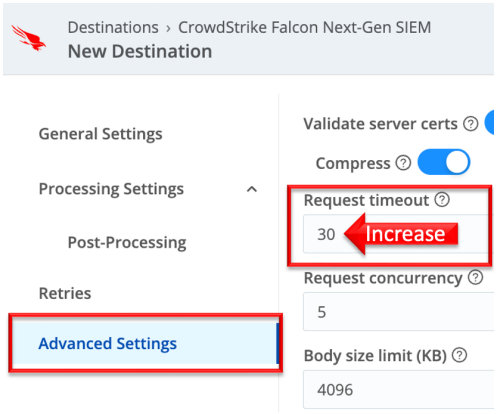
- **Next-Gen SIEM - Advanced Settings – Requestion timeout**: The 'Request timeout' configuration under the advanced settings of the Next-Gen SIEM destination can potentially cause duplications if Next-Gen SIEM does not response to Stream within the allotted time.
- **Output Router**: Similar to the Route configuration issues, Output Routers leverage a set of rules to determine where to route specific data and can be configure to be the 'final' evaluation of the data. The best way to prevent duplicates from being created with this Destination configuration is to ensure that each rule is configured to be 'Final' for the matching dataset.

Data is backing up or being dropped by Stream

- Check Stream's logs for a warning saying "Sender is at capacity". This indicates that the number of senders for the specific Next-Gen SIEM Stream Destination is too low for the amount of data that's being sent. In order to correct this, increase the "Request concurrency" number in the destination's "Advanced Settings".

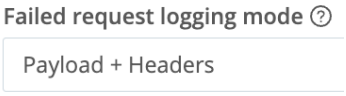


- Check Stream's logs for timeout errors. This would indicate that the communication stream is being closed before it's able to be completed. One way to attempt to correct this is to increase the "Request timeout" setting in the destination's "Advanced Settings".



API communication-based errors

Note: For the best troubleshooting experience ensure that the “Failed request logging mode” in the advanced configuration of the Next-Gen SIEM destination is configured to “Payload + Headers”



- In the event that there’s an issue with communicating the Next-Gen SIEM endpoint, begin by examining the Stream logs first. In the following example the API token was not input correctly, the header information provides key details for identifying issues such as this. It also provides a “trace-id” field value that needs to be provided to CrowdStrike support so that they can properly examine that communication.

A majority of authorization errors are caused by one or more of the following:

- Incorrectly entered API key in the "Next-Gen SIEM authentication token" configuration field in the Stream Destination configuration.
- Incorrectly entered connector URL in the "Next-Gen SIEM endpoint" configuration field in the Stream Destination configuration.
- Valid but mismatched values in the "Next-Gen SIEM authentication token" and "Next-Gen SIEM endpoint" configuration fields in the Stream Destination configuration.

Larger than expected data volumes in Next-Gen SIEM

Some misconfigurations within Stream can lead to larger than expected data volumes being sent to Next-Gen SIEM. In a majority of cases this involved the standard URL endpoint that's used by Next-Gen SIEM, the /services/collector endpoint. This is because the /services/collector endpoint provides the ability to include addition fields outside of the _raw event field in the Next-Gen SIEM event. The following are some examples of when event sizes may be larger in Next-Gen SIEM than they were when they were received by Stream.

- **Adding fields outside of the _raw field**: Adding fields to data going to a Next-Gen SIEM destination using the /services/collector endpoint will increase the overall size of the data being ingested. This is because additional fields that are added will be included in the 'Fields' field being sent to Next-Gen SIEM and will be included in the overall event data.
- **Renaming or copying the _raw field**: Renaming the _raw field (so that the event no longer has an _raw field) will result in Stream including the data in both the 'Event' and 'Fields' fields being sent to Next-Gen SIEM. This will not only produce much large Next-Gen SIEM event sizes but typically causes parsing errors as well. Copying the _raw field value to another field will create a field in the 'Fields' field that contains the same data and is the same size as the data that's in the 'Event' field, essentially doubling the overall event size. However, since the _raw field is still properly mapped to the 'Event' field there does not tend to be any issues with parsing.

Smaller event size\larger event counts than expected in Next-Gen SIEM

Some configurations can generate smaller than expect event sizes in conjunction with a larger overall number of events being generated in Next-Gen SIEM. This is often seen when sending formatted data, such as JSON data that has been formatted with indents and newlines, to the /services/collector/raw endpoint. The /services/collector/raw endpoint is not the default endpoint for Next-Gen SIEM connectors and should only be used when deemed necessary. The /services/collector/raw endpoint process the 'Event' field data only and processed each line as an independent event. As such, if the format was JSON with the proper indentation and line breaks then each line would be considered and processed as an independent event. NOTE: In most cases, because the event is incomplete, there will be parsing errors present.

Parsing Errors in Next-Gen SIEM

A common issue seen in Next-Gen SIEM is parsing errors. These errors can happen for a variety of reasons but there are some that are specific to when data is sent from Stream. The following are some examples of what can cause parsing issues when sending data from Stream.

- **Changing the _raw fields**: The _raw field in Stream is what is mapped to the 'Event' field being sent to Next-Gen SIEM. This is the field that will be processed by the parser and as such it needs to be in a format that the parser can handle and contain the necessary data that the parser is looking for. In some formats, such as JSON, adding or removing fields may not have any impact (as long as the necessary fields are still present). However, formats such as CSV are not as forgiving. For example, PaloAlto Firewall logs are parsed based on the syntax sequence that PaloAlto Networks publishes. This means that if a value is removed then the field mapping that takes place after the removed field will no longer be accurate.
- **The data version isn't currently supported by the parser**: Some parsers are configured to process specific log versions/types.
    - New (or even old) versions of events may have a different syntax and the parser is not able to properly parse that particular format. For example, PaloAlto Networks may add new fields to their events, modifying the CSV format to a point where the parser is no longer able to identify and associated each value to its proper field.
    - New events can be introduced into the data that haven't been included in the parser. In this case the events may be dropped by the parser or maybe posted with parsing errors.

- **The format the Stream sends is not the expected format**: In some cases, the data format that is send by Stream maybe different enough from the original source that the parser is not able to properly parse the data. For example, if a parser is configured to regex a 'pretty' JSON event (meaning a JSON event with indents and new line characters) but receives a 'flat' JSON event (meaning no indents or new line characters) it might not be possible to perform the regex actions. If this happens and the parser isn't able to properly regex the data, the parsing will most likely fail.

  If the parsing error(s) in Next-Gen SIEM are being created by parsers that are supplied and maintained by CrowdStrike please ensure that the data structure that's being provided is the same structure that the parser supports. This can often be done by doing an event capture in Stream of the _raw field data and testing it against a copy of the parser in Next-Gen SIEM. See the Testing Parsers section of this document for more information.