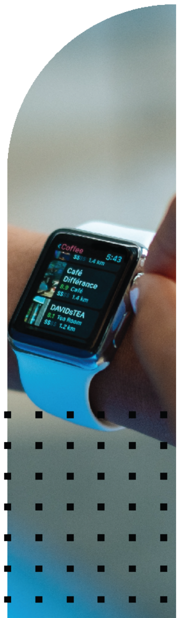
Connectors Guide

FortiSOAR 7.0.1



July, 2021
FortiSOAR 7.0.1 Connectors Guide
00-400-000000-20201228

TABLE OF CONTENTS

Change Log

Date	Change Description
2021-07-09	Initial release of 7.0.1

Introduction to connectors

Connectors are used to send and retrieve data from various third-party sources. Using connectors, you can connect to external cyber security tools and perform various automated interactions using FortiSOAR™ playbooks.

FortiSOAR has already developed a number of connectors that can be used to integrate with a number of external cyber security tools like SIEMs, such as Splunk and Ticketing systems such as Jira. Connector-specific documentation that is included with each connector covers the process of installing, configuring, and using these connectors. You can see a list of published connectors on the Fortinet Support Site.

FortiSOAR also provides you with a number of pre-installed connectors or built-ins that you can use within FortiSOAR playbooks, as a connector step, and perform automated operations. For more information on FortiSOAR Built-in connectors, see the "FortiSOAR Built-in connectors" article.

You can write a custom connector that can retrieve data from custom sources. You can then use the connector either standalone or within FortiSOAR playbooks and perform automated operations. You can write a custom connector manually, or you can use the FortiSOAR Connector SDK to develop your custom connector. For more information on writing custom connectors, see the Building your own connector chapter.

A Connector Store is part of the FortiSOAR UI. Use the Connector Store to easily view, search, install, update, and uninstall connectors that are part of the FortiSOAR repository. You can go to the connector store by clicking the **Connector Store** link that appears on the top-right corner on the `Connectors` page. For more information, see Connector Store. You can also specify connector configuration using a jinja variable that contains the connector configuration name.

Installing a FortiSOAR connector

All connectors provided by FortiSOAR are delivered using a FortiSOAR repository. Use the Connector Store to easily install, update and install connectors that are part of the FortiSOAR repository.

The recommended way to install a connector is by using the Connector Store.

To install a connector, you must be assigned a role that has a minimum of `Read` and `Create` access to the `Connectors` module.

You must ensure that update.cybersponse.com is reachable from your FortiSOAR instance. Otherwise, you will see a blank page when you click the **Connector Store** link.

You could also set up your FortiSOAR repository and use the `yum` command to install connectors as a *root* user:

```
# yum install cyops-connector-<connectorname>
```

After you install a connector using the `yum` command the new connector is not reflected until you refresh the `Connectors` page.

To update a FortiSOAR-provided connector use the following command:

```
# yum update cyops-connector-<connectorname>
```

To remove a FortiSOAR-provided connector use the following command:

```
# yum remove cyops-connector-<connectorname>
```

If you delete a FortiSOAR-provided connector using the Connector page in FortiSOAR then you cannot reinstall the RPM of the same connector because the RPM of the connector does not get deleted. Therefore, to remove a FortiSOAR-provided connector, you must use the `yum remove cyops-connector-<connectorName>` command.

Some of the connectors have dependencies on additional python packages. During connector installation, these dependencies are also installed using `pip`. The default pip settings point to the pypi.python.org repository for downloading these packages. We have also added an alternate repository on update.cybersponse.com to host the connector dependencies. If your instance restricts access to pypi.python.org, the installer fallbacks to this alternate repository for installing the dependencies. However, the connector installation might take longer if there are multiple dependencies since it first tries to fetch each of those from pypi.python.org. In such a case, it is recommended to switch to update.cybersponse.com as the main repository for the python packages also. You can switch the main repository to update.cybersponse.com so by editing the `pip.conf` file located at: `/opt/cyops-integrations/.env/pip.conf` as follows:
`index-url=`https://update.cybersponse.com/connectors/deps/simple/

You can also change other relevant pip settings such as altering the timeout or retry count setting. See https://pip.pypa.io/en/stable/user_guide/#config-file for a listing of the relevant pip settings. For example, to increase the timeout value add the following in the `pip.conf` file: `timeout = 60`.

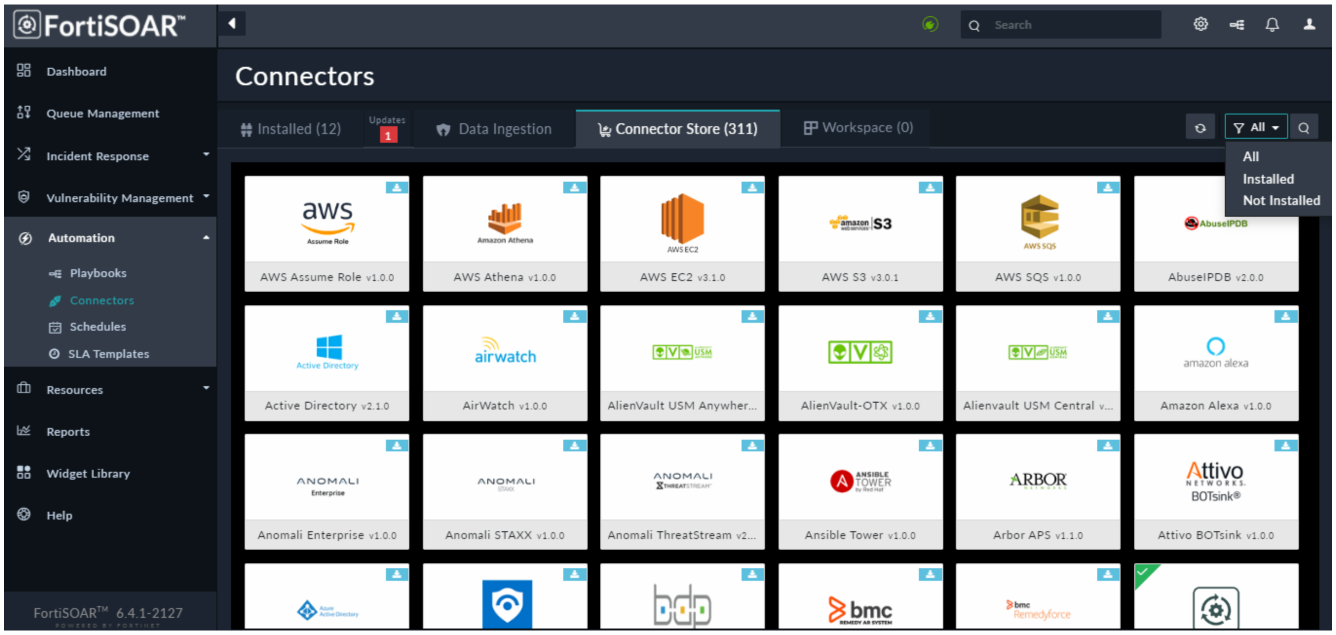
Connector Store

Use the Connector Store to easily view, search, install, upgrade, and uninstall connectors that are part of the FortiSOAR repository. Therefore, you can now perform these operations using the FortiSOAR UI instead of required CLI access.

Following are the permissions that you must be assigned to perform operations on connectors:

- To install a connector, you must be assigned a role that has a minimum of `Create` and `Read` access on the `Connectors` module and `Read` access on the `Playbooks` module.
- To upgrade or configure a connector, you must be assigned a role that has a minimum of `Update` and `Read` access on the `Connectors` module and `Read` access on the `Playbooks` module.
- To uninstall a connector, you must be assigned a role that has a minimum of `Delete` and `Read` access on the `Connectors` module and `Read` access on the `Playbooks` module.
- To view connectors and to use the connector as a step in the playbooks, you must be assigned a role that has a minimum of `Read` access on the `Connectors` and `Playbooks` module.

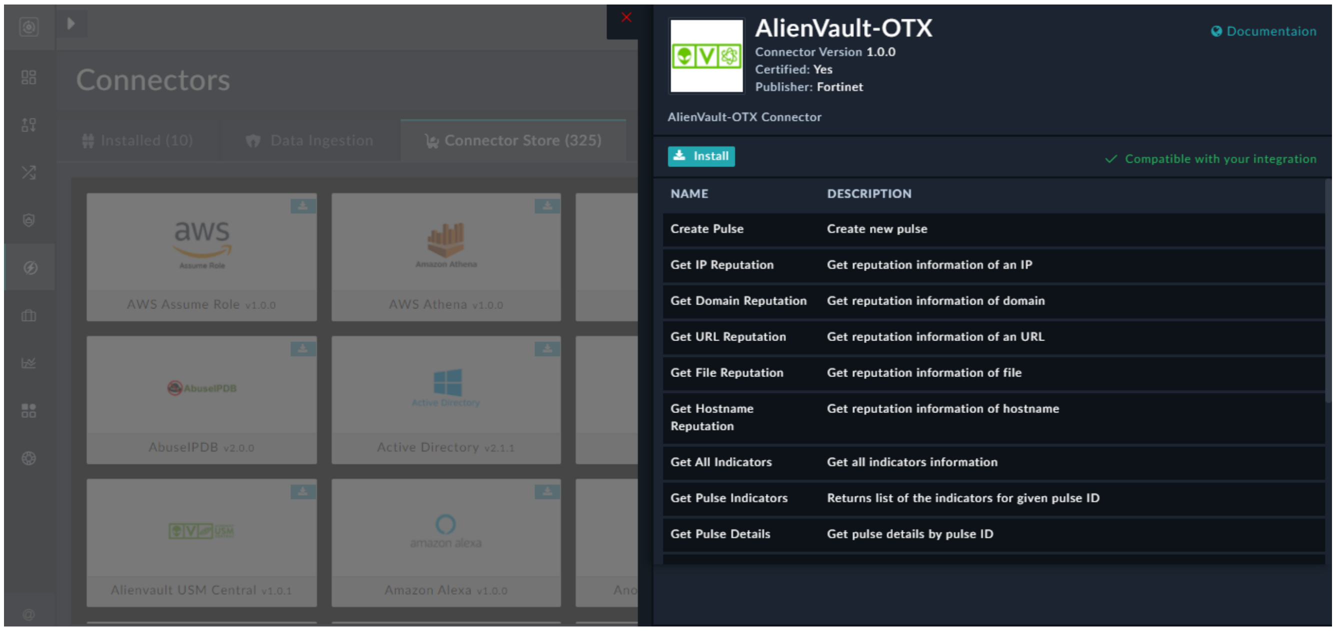
To go to the connector store, click **Automation** > **Connectors**. On the `Connectors` page, click the **Connector Store** tab. The `Connector Store` page appears as shown in the following image:



You can search for a connector by its name in the **Search by connector name** field.

The `Connector Store` page contains a filter for installed and not installed connectors. You can filter connectors by clicking the **Filter** drop-down list and choosing between **All**, **Installed** or **Not Installed** filters. The chosen filter applies only to the `Connector Store` page.

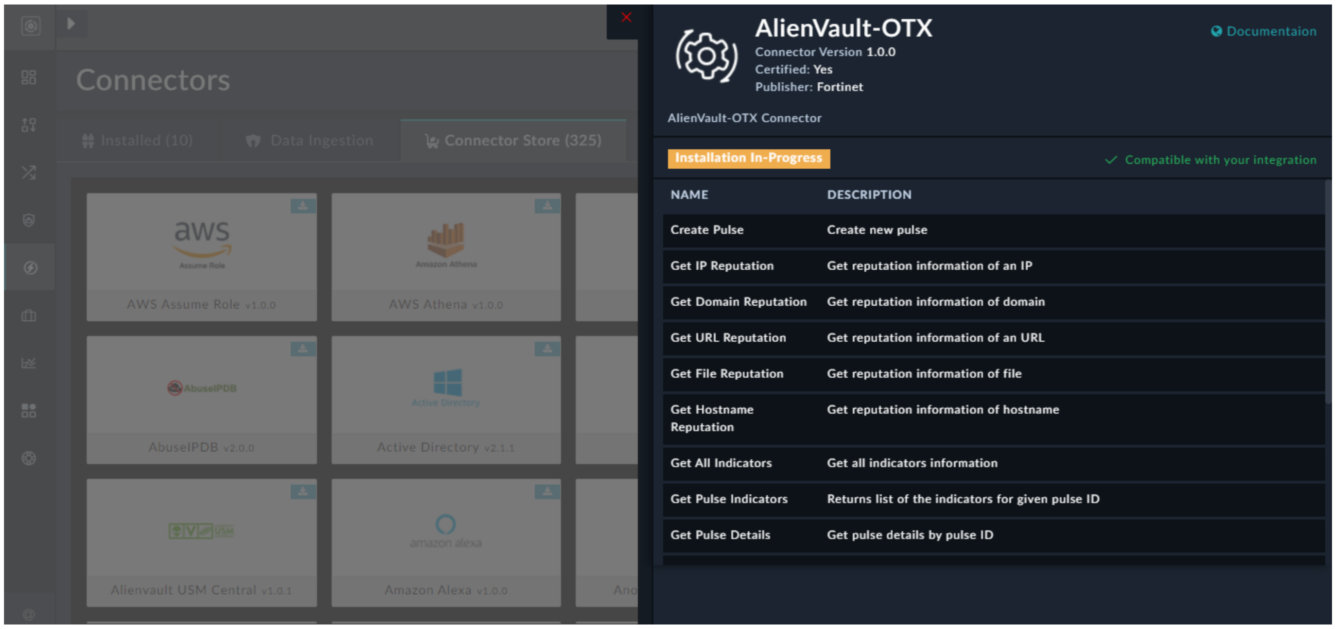
Connectors that you can install appear with an **Install** (blue) icon as shown in the above image. To install a connector, click the connector card of the connector that you want to install, for example, AlienValult OTX, which opens a popup with the connector name:


AlienVault-OTX
Connector Version 1.0.0
Certified: Yes
Publisher: Fortinet
Documentaion
AlienVault-OTX Connector
Install
Compatible with your integration

NAME	DESCRIPTION
Create Pulse	Create new pulse
Get IP Reputation	Get reputation information of an IP
Get Domain Reputation	Get reputation information of domain
Get URL Reputation	Get reputation information of an URL
Get File Reputation	Get reputation information of file
Get Hostname Reputation	Get reputation information of hostname
Get All Indicators	Get all indicators information
Get Pulse Indicators	Returns list of the indicators for given pulse ID
Get Pulse Details	Get pulse details by pulse ID

The connector popup contains details such as, a brief description of the connector, whether the connector is certified or not, who is the publisher of the connector, a list of actions the connector can perform, link to the connector documentation, etc.

Click **Install** in the connector name popup to begin the installation of the connector, as shown in the following image:



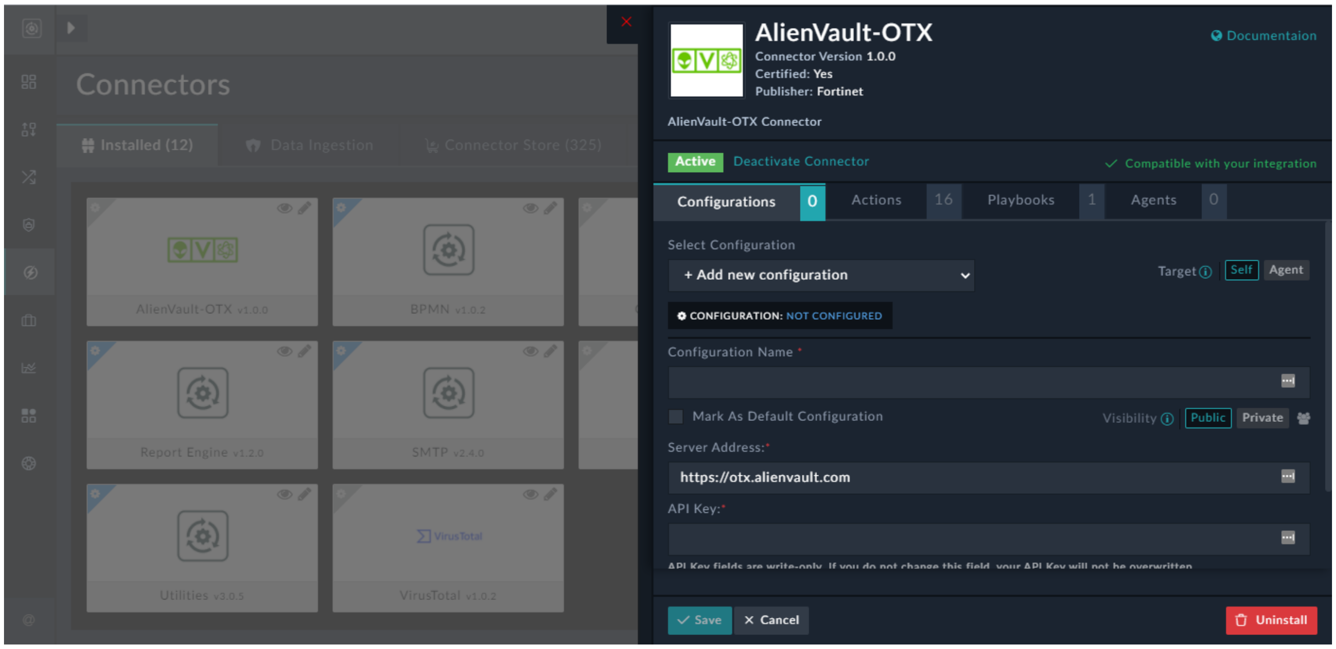
AlienVault-OTX
Connector Version 1.0.0
Certified: Yes
Publisher: Fortinet

AlienVault-OTX Connector

Installation In-Progress ✓ Compatible with your integration

NAME	DESCRIPTION
Create Pulse	Create new pulse
Get IP Reputation	Get reputation information of an IP
Get Domain Reputation	Get reputation information of domain
Get URL Reputation	Get reputation information of an URL
Get File Reputation	Get reputation information of file
Get Hostname Reputation	Get reputation information of hostname
Get All Indicators	Get all indicators information
Get Pulse Indicators	Returns list of the indicators for given pulse ID
Get Pulse Details	Get pulse details by pulse ID

Once installation is complete, FortiSOAR displays the "Connector Installed successfully" message, `Active` is displayed on the popup, and on the main connectors page you will see the installed connectors number increase by 1. After installing the connector, you must configure the connector by entering the required configuration details in the connector popup:



AlienVault-OTX
Connector Version 1.0.0
Certified: Yes
Publisher: Fortinet

AlienVault-OTX Connector

Active ✓ Compatible with your integration

Select Configuration: + Add new configuration

Target: Self | Agent

CONFIGURATION: NOT CONFIGURED

Configuration Name *

Mark As Default Configuration

Visibility: Public | Private

Server Address:* https://otx.alienvault.com

API Key:*

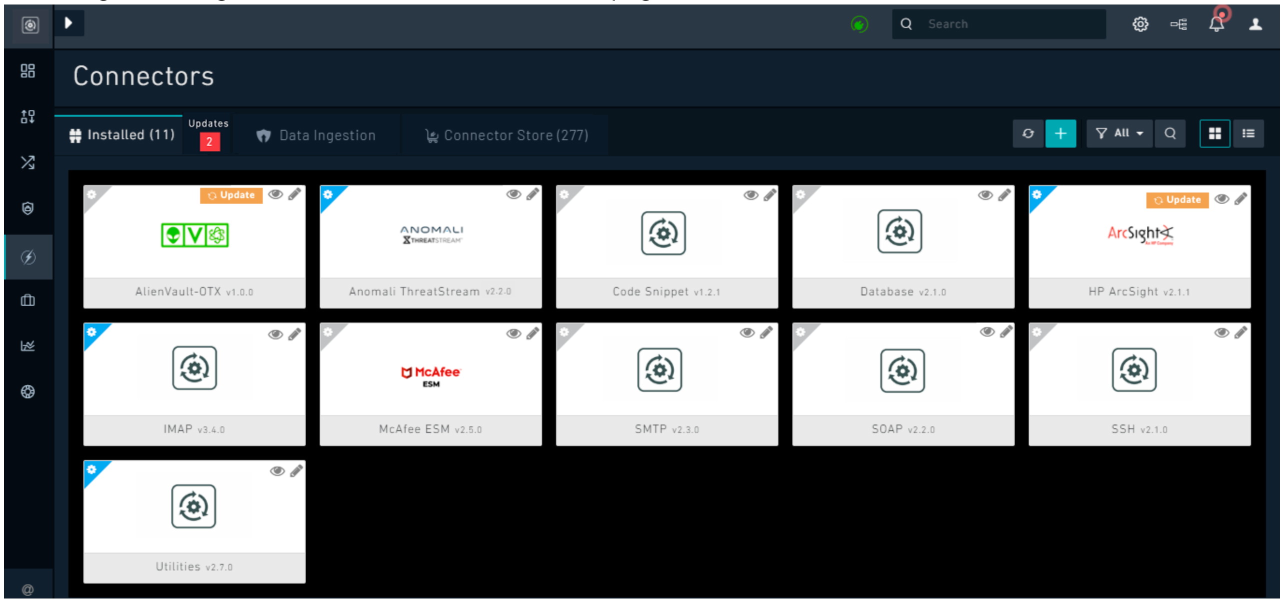
API Key fields are write-only. If you do not change this field, your API Key will not be overwritten.

You can also configure the connector on the `Connectors` page by clicking the connector name card, which will also display the same connector popup.

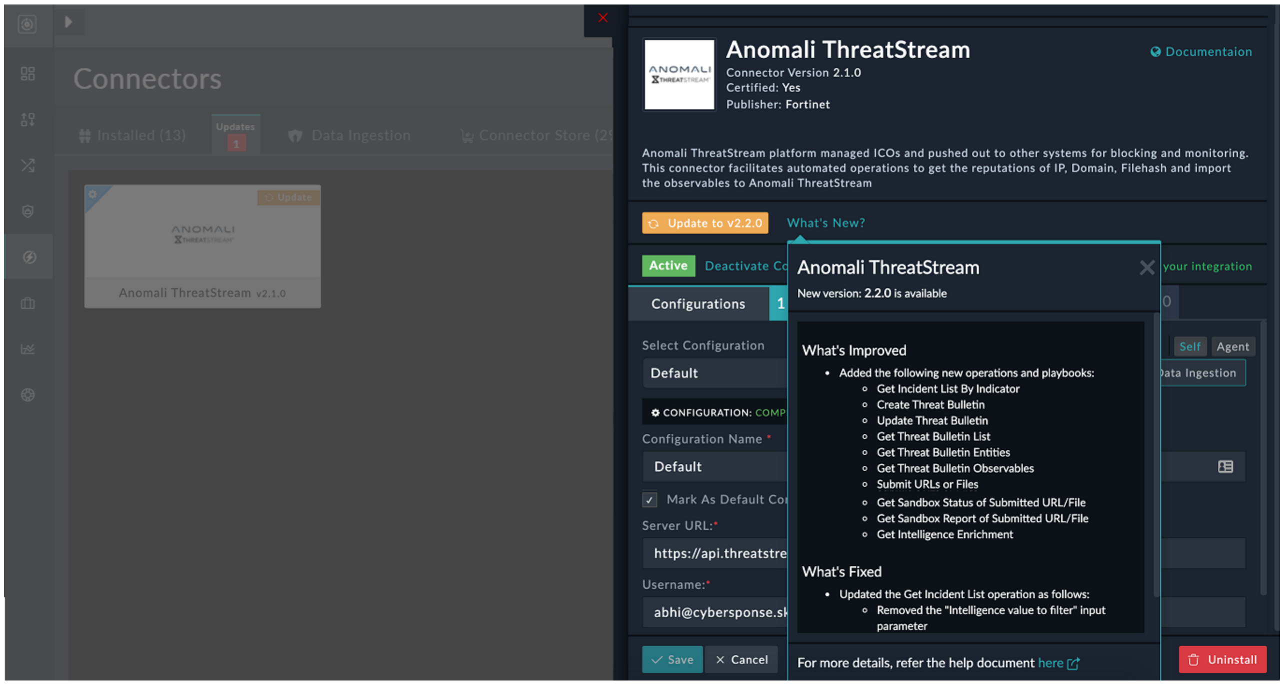
To uninstall a connector, click the **Uninstall Connector** icon, in the connector popup. FortiSOAR displays a Confirmation dialog, click **Confirm** to uninstall the connector. FortiSOAR displays the "Connector uninstalled successfully" message and the installed connectors number decreases by 1.

Installing or importing and configuring a connector in FortiSOAR

Use the Connector Store to install and configure connectors in FortiSOAR. To install a connector, you must be assigned a role that has a minimum of `Create` access to the `Connectors` module. To configure connectors into FortiSOAR, you must be assigned a role that has a minimum of `Update` access to the `Connectors` module.

1. Log on to FortiSOAR.
2. On the left navigation pane, click **Automation** > **Connectors** > **Installed**.
   On `Installed Connectors` page, you will see the list of installed connectors, either in grid/list view or in the card view. You will also see a **Data Ingestion** tab on the `Connectors` page, which displays the connectors that are enabled for the data ingestion wizard. For information on the Data Ingestion tab, see the Data Ingestion chapter.
   Following is an image of the `Installed Connector` page in the Card view:
   

   In the top bar of the `Connectors` page, you can see the number of connectors that are installed, for example, in the above image 11 connectors are installed. In the top bar, you can also view if any connector that is installed and configured on your system has an upgraded version, for example, if you have version 2.1.0 of the Anomali ThreatStream connector installed on your system and Fortinet has released a newer version of this connector, i.e., version 2.2.0 of the Anomali ThreatStream connector, and similarly there is an update to the HP ArcSight connector also, then the Updates button will display **2**. To update a connector that you have installed and configured on your system, click the **Updates** button, which will display the connectors with the updated version. You can choose to then install the updated connector by clicking on the connector card and then clicking the **Update to 'version'** button on the connector configurations page. From version 6.4.1 onwards, when you click on a connector that has an updated version, you will see a '**What's new?**' section on the configuration page, besides the **Update to version** button. Clicking the **What's New?** link displays the highlights of the new version, which you can view to understand what you can expect when you upgrade your connector version.



Anomali ThreatStream
Connector Version 2.1.0
Certified: Yes
Publisher: Fortinet

Anomali ThreatStream platform managed ICOs and pushed out to other systems for blocking and monitoring. This connector facilitates automated operations to get the reputations of IP, Domain, Filehash and import the observables to Anomali ThreatStream

Update to v2.2.0 What's New?

Active Deactivate

Select Configuration: Default
CONFIGURATION: COMP
Configuration Name *: Default
✓ Mark As Default Co
Server URL:*: https://api.threatstre
Username:*: abhi@cybersponse.sk

Anomali ThreatStream
New version: 2.2.0 is available

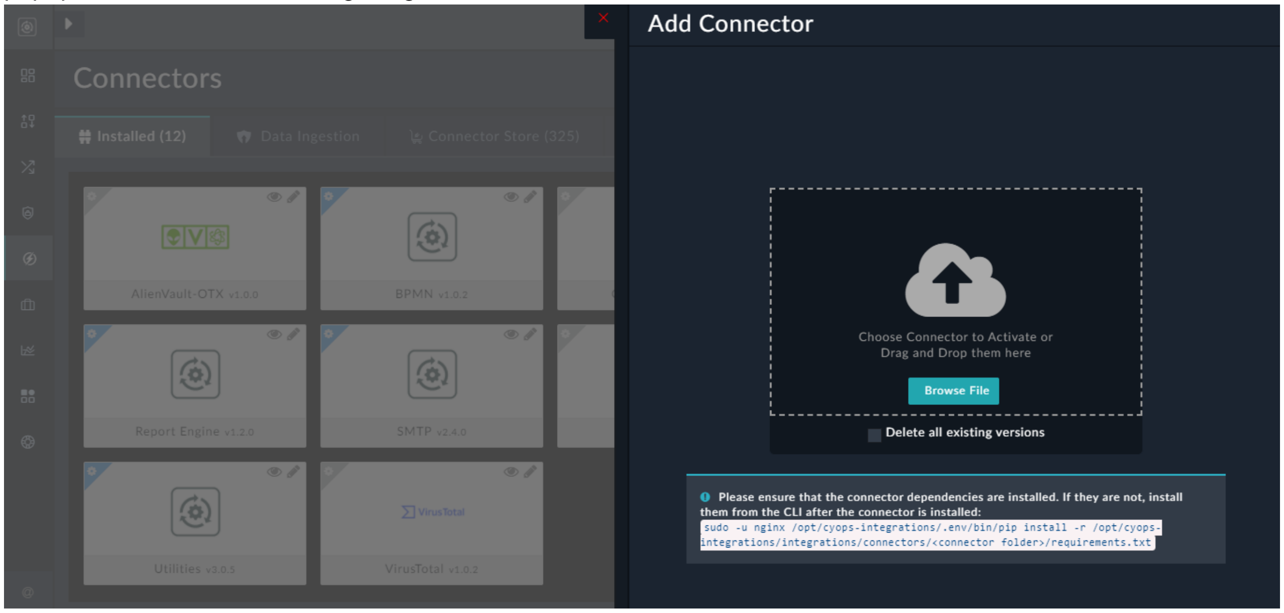
What's Improved
- Added the following new operations and playbooks:
  - Get Incident List By Indicator
  - Create Threat Bulletin
  - Update Threat Bulletin
  - Get Threat Bulletin List
  - Get Threat Bulletin Entities
  - Get Threat Bulletin Observables
  - Submit URLs or Files
  - Get Sandbox Status of Submitted URL/File
  - Get Sandbox Report of Submitted URL/File
  - Get Intelligence Enrichment

What's Fixed
- Updated the Get Incident List operation as follows:
  - Removed the "Intelligence value to filter" input parameter

For more details, refer the help document here

Also, you can see the status of the connector, based on the icon present on the top-left of the connector card. A connector that is installed but not configured appears with a **Settings** icon on a gray background, for example, the AlienVault-OTX connector. A connector that is installed and configured, for example the HP ArcSight connector, or a connector that does not require any configuration, for example the Utilities connector, appears with a Settings icon on a blue background.

Click **+** > **Upload Connector** to import a connector (`.tgz`) file into FortiSOAR, which displays the **Add Connector** popup as shown in the following image:



Add Connector

Choose Connector to Activate or Drag and Drop them here

Delete all existing versions

Please ensure that the connector dependencies are installed. If they are not, install them from the CLI after the connector is installed:

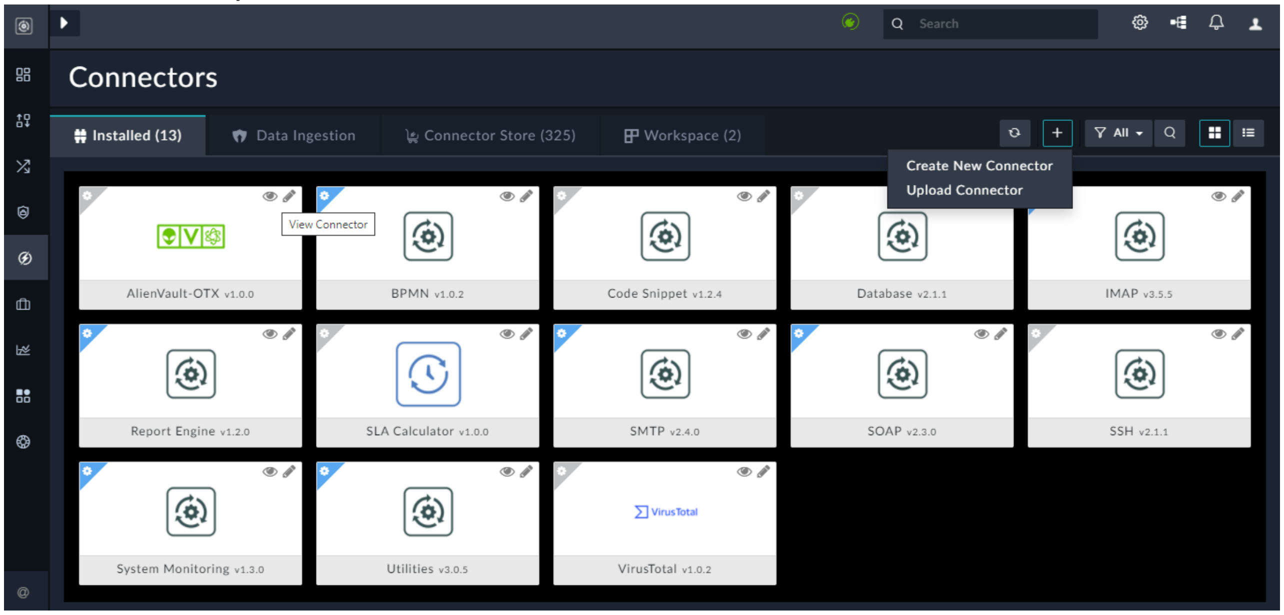
```
sudo -u nginx /opt/cyops-integrations/.env/bin/pip install -r /opt/cyops-integrations/integrations/connectors/<connector folder>/requirements.txt
```

You can drag-and-drop the connector .tgz file onto the popup or browse to the .tgz file to install the connector in FortiSOAR. You must check that all the connector dependencies are available or install the dependencies additional to ensure that the connector functions as expected.

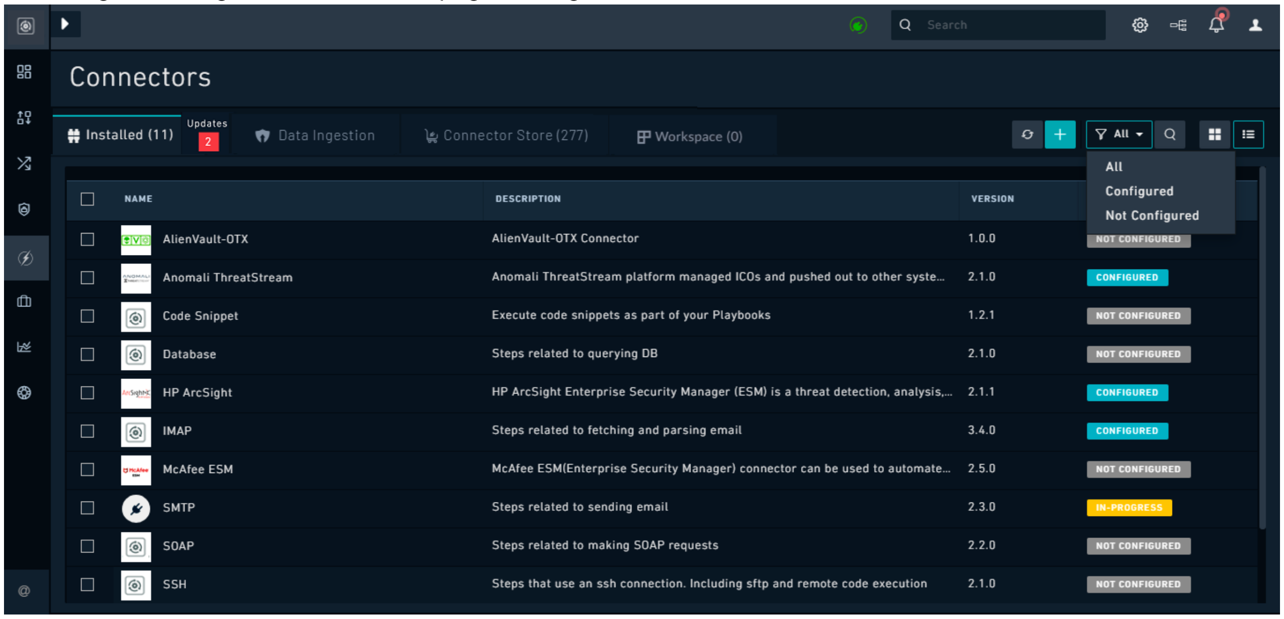
Note: In case your FortiSOAR configuration is a High Availability (HA) cluster, then if you install a connector using a `.tgz` file, you will need to install it separately on all nodes of the HA cluster.

If you want to create a new connector, click **+** > **Create New Connector**. For more information on creating new

connectors, see the Building your own connector chapter.
You can search for a connector by connector name in the **Search by connector name** box. You can also filter connectors by clicking the **Filter** drop-down list and choosing between **All**, **Configured** or **Not Configured** filters. The chosen filter applies only to the `Installed Connectors` page.
On a connector card you can also see **View Connector** and **Edit Connector** icons:



Clicking the **View Connector** icon opens the code editor interface for the selected connector. The folder structure is the similar to the folder structure defined in the Building your own connector chapter.
You can make updates to a connector according to your requirements by clicking the **Edit Connector** icon; however, to edit a connector, you must make a local copy of the connector and then make the changes in the same, and then you can save and publish the connector as defined in the Building your own connector chapter.
Buttons to change the view from grid to card and vice-versa are present on the right of the `Connectors` page.
Following is an image of the Connector page in the grid view:

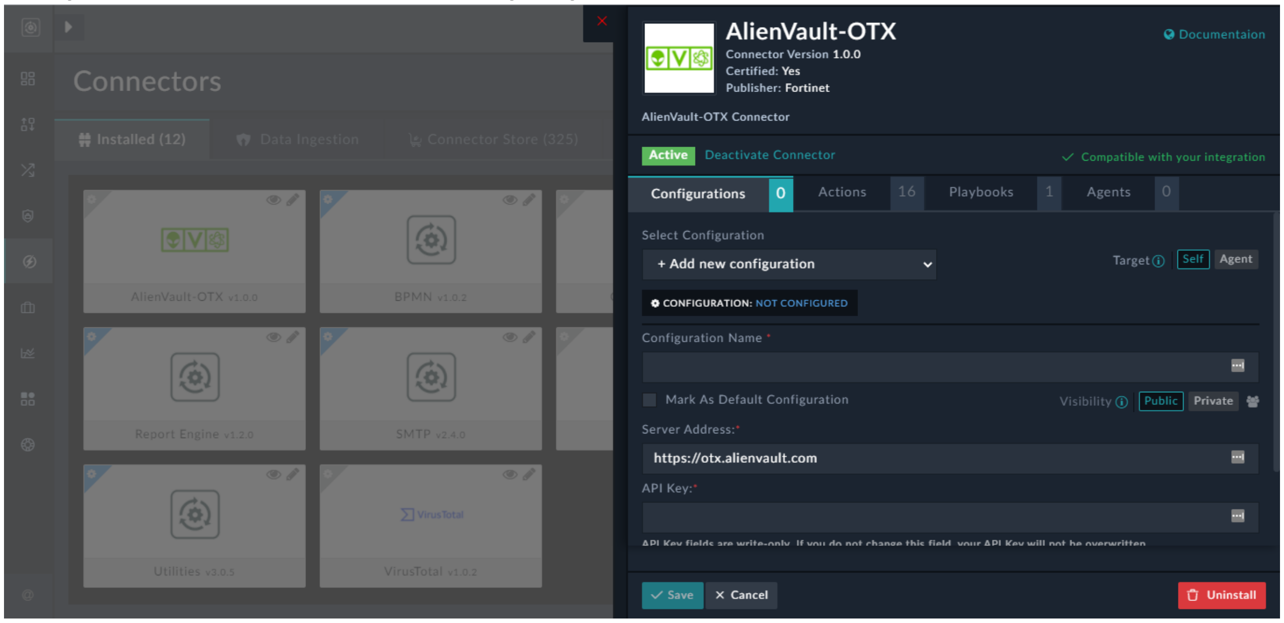


NAME	DESCRIPTION	VERSION	
AlienVault-OTX	AlienVault-OTX Connector	1.0.0	NOT CONFIGURED
Anomali ThreatStream	Anomali ThreatStream platform managed ICOs and pushed out to other syste...	2.1.0	CONFIGURED
Code Snippet	Execute code snippets as part of your Playbooks	1.2.1	NOT CONFIGURED
Database	Steps related to querying DB	2.1.0	NOT CONFIGURED
HP ArcSight	HP ArcSight Enterprise Security Manager (ESM) is a threat detection, analysis,...	2.1.1	CONFIGURED
IMAP	Steps related to fetching and parsing email	3.4.0	CONFIGURED
McAfee ESM	McAfee ESM(Enterprise Security Manager) connector can be used to automate...	2.5.0	NOT CONFIGURED
SMTP	Steps related to sending email	2.3.0	IN-PROGRESS
SOAP	Steps related to making SOAP requests	2.2.0	NOT CONFIGURED
SSH	Steps that use an ssh connection. Including sftp and remote code execution	2.1.0	NOT CONFIGURED

You can see a list of the connectors with their associated brief descriptions, version installed and the status. The status of the connectors will be **Installed** for connectors that are installed but not configured, such as AlienVault-OTX, or **Configured** for connectors that are installed and configured, such as HP ArcSight, or for connectors that do

not require any configuration, such as Utilities.
Click the **Connector Store** tab to view the connector store.

3. To configure a connector, click the connector row (if you are in the grid view) or the connector card (if you are in the card view) to open the **Connector Configuration** popup. Enter the required configuration details in the Connector Configuration popup, as shown in the following image:

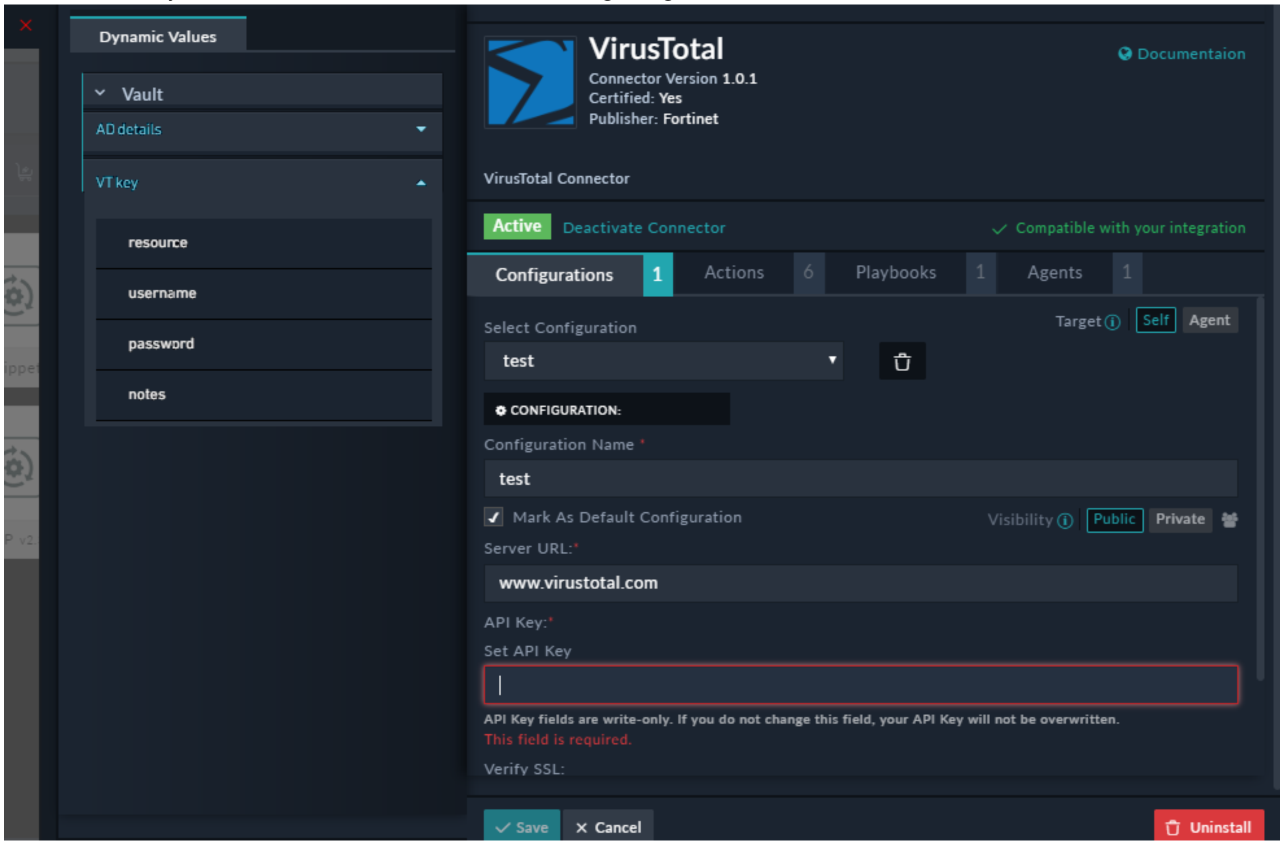


AlienVault-OTX
Connector Version 1.0.0
Certified: Yes
Publisher: Fortinet
AlienVault-OTX Connector

Select Configuration: + Add new configuration
Target: Self | Agent
CONFIGURATION: NOT CONFIGURED
Configuration Name *
Mark As Default Configuration
Visibility: Public | Private
Server Address:* https://otx.alienvault.com
API Key:*
API Key fields are write-only. If you do not change this field, your API Key will not be overwritten.

You can uninstall a connector by clicking the **Uninstall Connector** icon. To uninstall a connector, you must be assigned a role that has a minimum of `Delete` access to the `Connectors` module.
From version 6.4.1 onwards, you can configure connectors for the current FortiSOAR node, an agent, which is used to remotely run actions, or both. For more information on agents and how to run remote actions using agents, see the *Segmented Network support in FortiSOAR* chapter in the "Administration Guide." You can configure on the current node (Self) or on a remote Agent node (Agent) by clicking the **Self**, which is the default, or **Agent** buttons besides `Target`. You can configure multiple configurations for a connector on both the current node and the agent node.
To configure connector on the "Self" node, click **Self**, which is the default, if you are configuring the connector for the first time or if you want to add a new configuration, then click **Add new configuration** from the **Select Configuration** drop-down list and then add the name of the configuration and specify the configuration parameters. If you want to update an existing configuration, then select the configuration from the **Select Configuration** drop-down list and update the configuration parameters. If there is only one configuration, then that configuration will be selected automatically. To configure and execute connector actions on the "Agent" node, click **Agent**, and from the **Select Agent** drop-down list select the agent on which you want to run the connector actions. If there is only one agent installed, then that agent will be selected automatically. If you are configuring the connector for the first time or if you want to add a new configuration, then click **Add new configuration** from the **Select Configuration** drop-down list and then add the name of the configuration and specify the configuration parameters. If you want to update an existing configuration, then select the configuration from the **Select Configuration** drop-down list and update the configuration parameters. If there is only one configuration, then that configuration will be selected automatically.
You can also define the ownership of connector configurations by setting the visibility of a connector's configuration to 'Private'. For more information, see the Defining the ownership of a connector's configuration section.
Note: You can add multiple configurations for your connector if you have more than one instance of your third-party server in your environment. You must, therefore, add a unique `Name` for each configuration in the **Configuration Name** field.
If you have previous versions of a connector and you are configuring a newer version of that connector, with the same configuration parameters, then FortiSOAR fetches the configuration and input parameters of the latest available version of that connector. For example, if you have 1.0.0 and 2.0.0 versions of the Database connector

and you are configuring the 2.0.0 version of the Database connector, then while configuring the 2.0.0 version, FortiSOAR will fetch the configuration and input parameters from the 1.0.0 version of the Database connector. You can review the configuration and input parameters, and then decide to change them or leave them unchanged.
You can activate or deactivate a configured connector by clicking on the **Activate Connector** or **Deactivate Connector** Link.
You can also check the **Mark As Default Configuration** option to make the selected configuration, the default configuration of this connector, on the particular FortiSOAR instance. This connector will point to this configuration by default.
Important: In the case of the SMTP connector, you must ensure that this option is selected for the configuration that is to be used for sending system notifications.
The `password` type fields in FortiSOAR include encryption and decryption. Passwords are encrypted before saving them into the database and decrypted when they are used in actions. In case of an upgrade, connectors that are already installed will work with stored passwords. If your administrator has configured an external vault to securely store your organization's sensitive data and credentials, then you can use the Dynamic Values dialog to enter the credentials for your connector as shown in the following image:



Connector Version 1.0.1
Certified: Yes
Publisher: Fortinet

VirusTotal Connector

Select Configuration: test

Configuration Name *: test

✓ Mark As Default Configuration

Server URL:*: www.virustotal.com

API Key:*
Set API Key

API Key fields are write-only. If you do not change this field, your API Key will not be overwritten.
This field is required.

Verify SSL:

For information on Dynamic Values, see the *Dynamic Values* chapter in the "Playbooks Guide." For information on Password Vault, see the *Security Management* chapter in the "Administration Guide."
Connectors also include a **Verify SSL** field, that specifies whether the SSL certificate for the server is to be verified or not. By default, this option is set as `True`. For more information, see How the connector framework verifies the server certificate when it's self-signed.
To view the documentation associated with a connector, click the **Documentation** link on the top-right corner of the connector configuration pane.

4. To save your configuration, click **Save**.
To view the list of actions that can be performed by the connector, click the **Actions** tab. On the Actions tab, you can also restrict specific connector actions to specific roles, i.e., users with only those roles would be able to perform that action. For more information, see the Restricting specific connector actions to specific roles section,
To view the playbook file that is bundled with the connector, click the **Playbooks** tab.
To view the list of agents on which the connector is installed, click the **Agents** tab. For more information on agents

and how to run remote actions using agents, see the *Segmented Network support in FortiSOAR* chapter in the "Administration Guide."

5. (Optional) You can optionally perform a Health Check by clicking the **Refresh** icon that is present in the `Health Check` bar. The Health Check checks if the configuration parameters you have specified are correct and if connectivity can be established to the specified server, endpoint or API.
   If all the details are correct and the connectivity to the server can be established, then on the Connectors page, **Available** is displayed in the health check dialog.
   If any or all the details are incorrect or if the connectivity to the server cannot be established then on the Connectors page, **Disconnected** is displayed in the health check dialog.

Points to be considered for connector configurations while upgrading to a newer version of the connector

If you are upgrading a connector to a newer version, you must be assigned a role that has a minimum of `Upgrade` access to the `Connectors` module. For example, if you are upgrading the Symantec Security Analytics connector version from v1.0.0 to v2.0.0, then keep a note of the following points:

- Existing (older) connector configuration fields retain their value, i.e., the value from the older configuration will be displayed in the configuration pane of the newer connector version. New connector configuration field(s), if any, will be added to the connector configuration pane.
- If the newly added configuration field is mandatory, and FortiSOAR has specified its default value (in the `info.json` file of the connector), then the configuration pane of the newer version of the connector will contain the default value for this configuration field. For more information on the connector framework and the `info.json` file, see the Building your own connector chapter.
  For information on common connector framework issues, see the *Common connector framework errors* section in the Debugging common playbook and connector issues article present in the Fortinet Knowledge Base.
- If the newly added configuration field is mandatory, and FortiSOAR has not specified its default value (in the `info.json` file of the connector), then the configuration pane of the newer version of the connector will contain a `blank` value for this configuration field. If you also do not specify a value for this mandatory configuration field, then the connector configuration pane will display **Partially Configured**, and an error will also be displayed in the Playbook Execution Log. For more information on the Playbook Execution Log, see the *Debugging and Optimizing Playbooks* chapter in the "Playbooks Guide."
- If the field type of a mandatory configuration field is changed from the older version to the newer version, for example from a text field to a drop-down list, then the value of that field will ***not*** be retrieved from the older version. However, if FortiSOAR has specified its default value (in the `info.json` file of the connector), then that value will be displayed for this configuration field the configuration pane of the newer version of the connector. If however FortiSOAR has not defined the default value and you also do not specify a value for this mandatory configuration field, then the configuration pane of the newer version of the connector will contain a `blank` value for this configuration field, and the connector configuration pane will display **Partially Configured**. An error will also be displayed in the Playbook Execution Log. For more information on the Playbook Execution Log, see the *Debugging and Optimizing Playbooks* chapter in the "Playbooks Guide."
- If the newly added configuration field is optional, and FortiSOAR has specified its default value (in the `info.json` file of the connector), then the configuration pane of the newer version of the connector will contain the default value for this configuration field. If there is no default value is set, then its value is set as `blank`.

How the connector framework verifies the server certificate when its self-signed

The connector framework is explained in the Building your own connector chapter.

All connector calls are made by the python requests library reading the certificate from `/opt/cyops-integrations/.env/lib/python3.6/site-packages/certifi/cacert.pem`. Therefore, for any connector, when you set `verify_ssl` to `true`, and it's a self-signed cert, then the `cacert` must be appended to this file. If it's a chain of trust, then you must add the entire chain in the `pem` format. You must also ensure that the server address added in the connector configuration matches the CN in the certificate.

A `.key` file has the path to a PEM encoded file containing the private key. A `.pem` file has the path to a PEM encoded file containing the certificate (or certificate chain) that will be presented when requested.

If you are using the HTTPS proxy for external connections, then you must ensure that proxy certificate is added here also, if the `Verify SSL` is set to true in the connector configuration.

Some commands that you can use to get the `pem` certificate chain:

```
# openssl s_client -connect {HOSTNAME}:{PORT} -showcerts
```
OR
If you have the certificate already in a `.crt`, `.cer`, `.der` format, then you need to convert to the `pem` format: `# openssl x509 -inform der -in certificate.cer -out certificate.pem`

Role-based Access Control for connector actions

From version 7.0.0 onwards, FortiSOAR introduces Role-based Access Control (RBAC) for connectors, i.e., 'Connector Administrators', i.e., users with Connector Read and Update permissions and Security Read permission can allow access to only certain teams or users, based on roles, to perform certain connector actions. For example, the administrator might want to allow a "Block IP" action to be performed by only certain teams or users in the organization.

The ownership of connector configurations can also be defined, by marking the connector configuration as 'Private'; thereby, controlling who can view and execute that particular connector configuration.

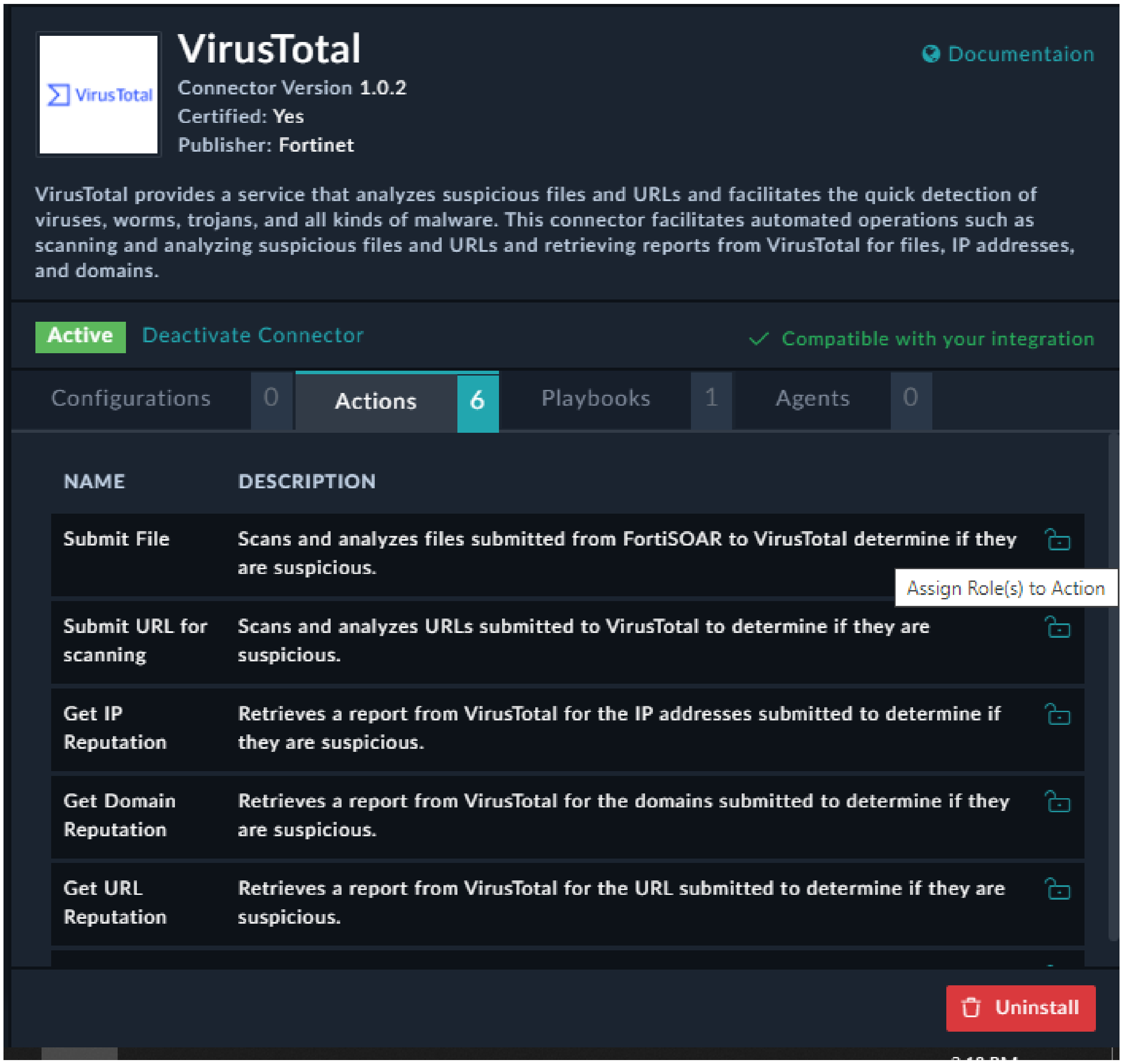
You must configure access control for connector configurations and actions separately for the tenant nodes. Access control for connector configurations and actions cannot be configured from the master.

Restricting specific connector actions to specific roles

By default, all users and roles with appropriate permissions, as defined in the Connector Store section, can view and execute all the connector actions, however connector administrators can restrict specific actions to specific roles as follows:

1. Log on to FortiSOAR.
2. On the left navigation pane, click **Automation** > **Connectors** > **Installed**.
3. Select the connector whose connector actions you want to restrict to certain users, based on the roles or specific set of roles that have been assigned to open the `Connector Configurations` page.

4. Click the **Actions** tab:



VirusTotal

Connector Version 1.0.2

Certified: Yes

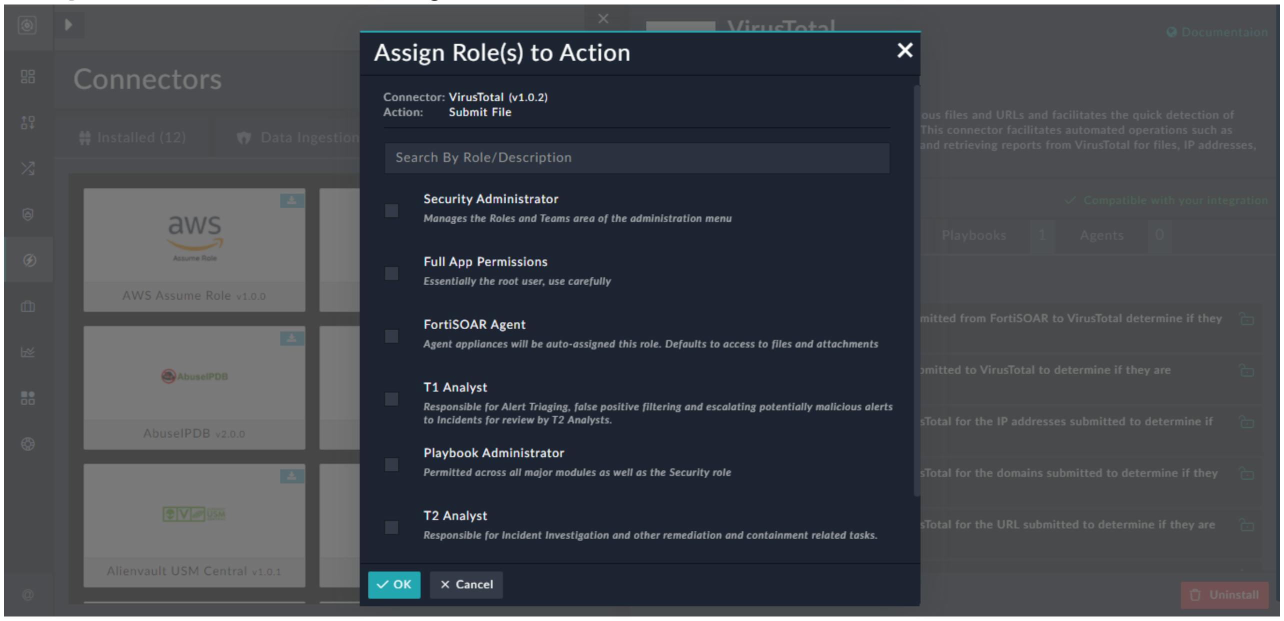
Publisher: Fortinet

Documentaion

VirusTotal provides a service that analyzes suspicious files and URLs and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware. This connector facilitates automated operations such as scanning and analyzing suspicious files and URLs and retrieving reports from VirusTotal for files, IP addresses, and domains.

NAME	DESCRIPTION
Submit File	Scans and analyzes files submitted from FortiSOAR to VirusTotal determine if they are suspicious.
Submit URL for scanning	Scans and analyzes URLs submitted to VirusTotal to determine if they are suspicious.
Get IP Reputation	Retrieves a report from VirusTotal for the IP addresses submitted to determine if they are suspicious.
Get Domain Reputation	Retrieves a report from VirusTotal for the domains submitted to determine if they are suspicious.
Get URL Reputation	Retrieves a report from VirusTotal for the URL submitted to determine if they are suspicious.

5. In the row that contains the action that you want to restrict, click the **Assign Role(s) to Action** icon to display the `Assign Role(s) to Action` dialog:



Assign Role(s) to Action

Connector: VirusTotal (v1.0.2)
Action: Submit File

Search By Role/Description

Security Administrator
Manages the Roles and Teams area of the administration menu

Full App Permissions
Essentially the root user, use carefully

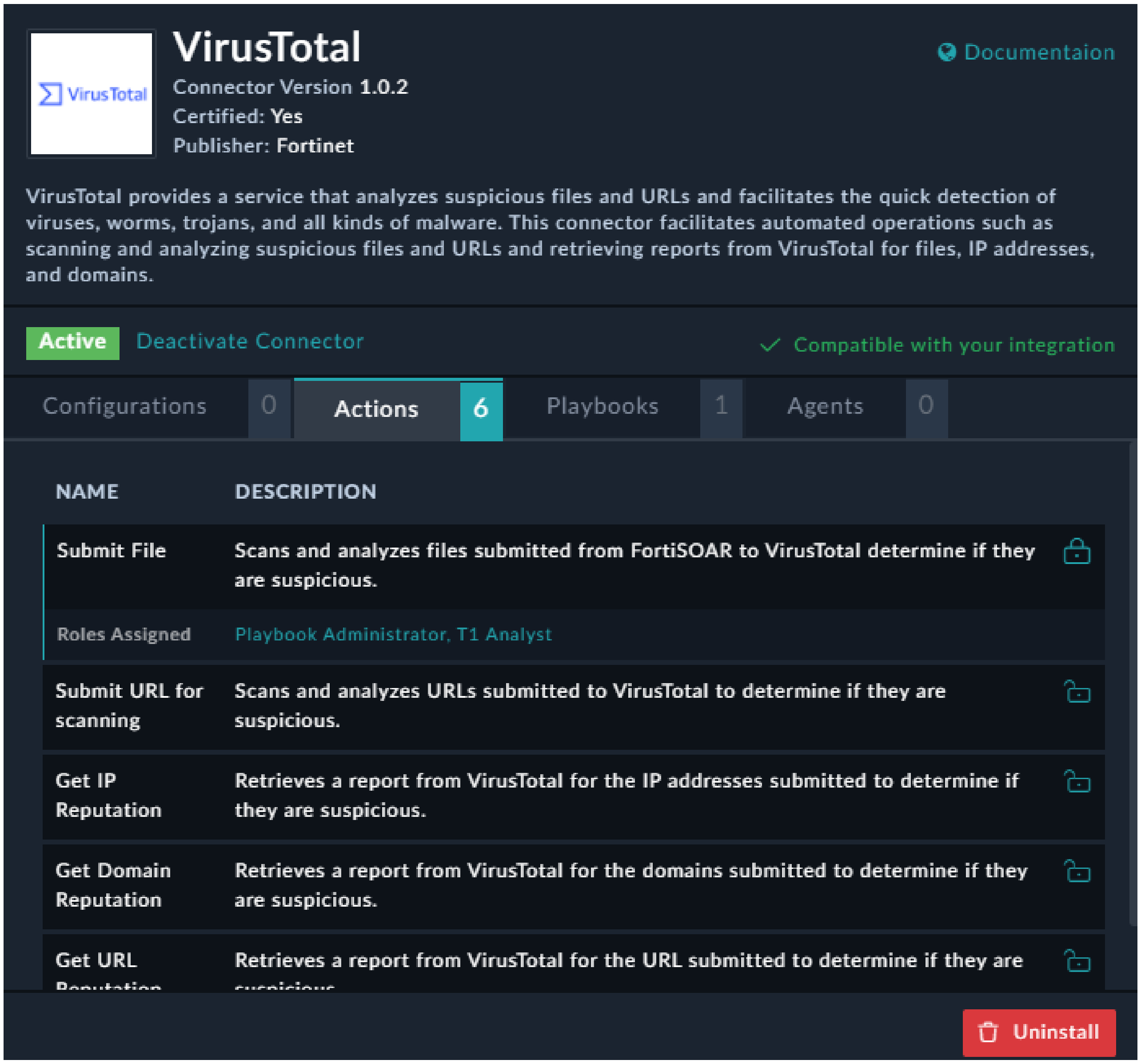
FortiSOAR Agent
Agent appliances will be auto-assigned this role. Defaults to access to files and attachments

T1 Analyst
Responsible for Alert Triaging, false positive filtering and escalating potentially malicious alerts to Incidents for review by T2 Analysts.

Playbook Administrator
Permitted across all major modules as well as the Security role

T2 Analyst
Responsible for Incident Investigation and other remediation and containment related tasks.

6. Select the roles that would be able to perform this action and then click **OK**.
Once a particular action is assigned to specific roles, you can see a **Roles Assigned** bar in that action's role, which specifies the roles that the users should be assigned to perform that connector action:



VirusTotal

Connector Version 1.0.2

Certified: Yes

Publisher: Fortinet

Documentaion

VirusTotal provides a service that analyzes suspicious files and URLs and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware. This connector facilitates automated operations such as scanning and analyzing suspicious files and URLs and retrieving reports from VirusTotal for files, IP addresses, and domains.

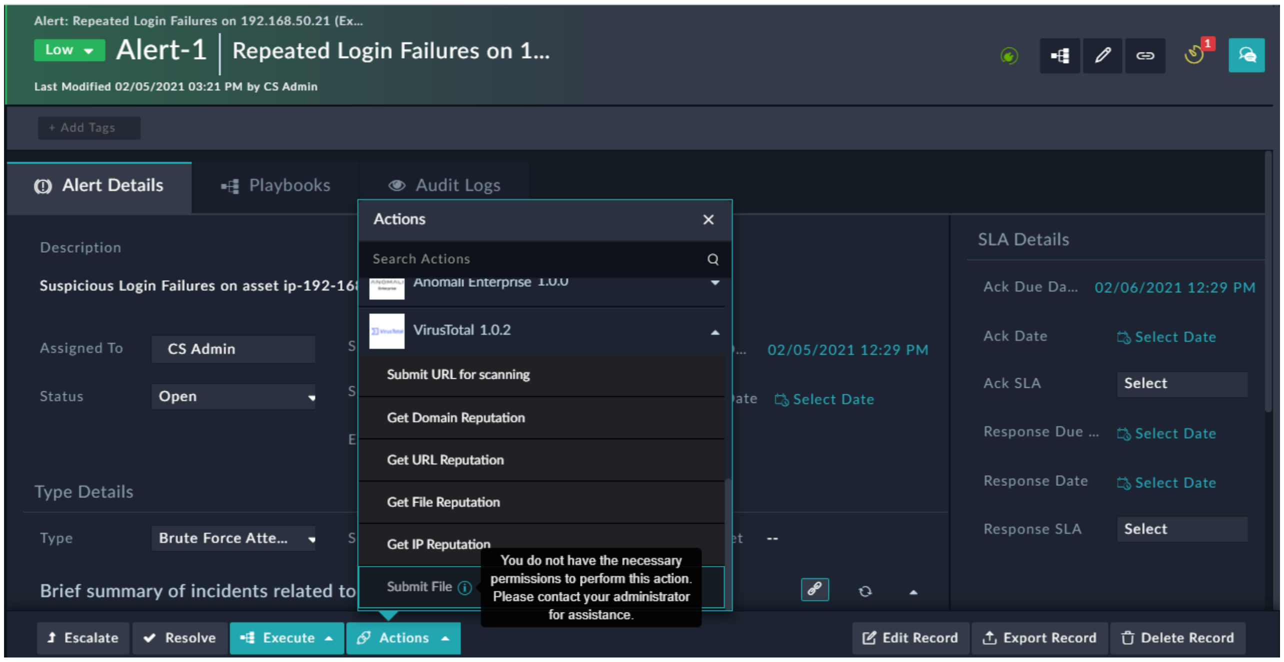
Active Deactivate Connector — Compatible with your integration

Configurations 0 | Actions 6 | Playbooks 1 | Agents 0

NAME	DESCRIPTION
Submit File	Scans and analyzes files submitted from FortiSOAR to VirusTotal determine if they are suspicious.
Roles Assigned	Playbook Administrator, T1 Analyst
Submit URL for scanning	Scans and analyzes URLs submitted to VirusTotal to determine if they are suspicious.
Get IP Reputation	Retrieves a report from VirusTotal for the IP addresses submitted to determine if they are suspicious.
Get Domain Reputation	Retrieves a report from VirusTotal for the domains submitted to determine if they are suspicious.
Get URL Reputation	Retrieves a report from VirusTotal for the URL submitted to determine if they are suspicious

Uninstall

Therefore, in this case if there are users who are not assigned the 'Playbook Administrator' or 'T1 Analyst' role, and if they try to invoke this action directly on a record, they will be unable to run that action as the action will be disabled:



Notes:

- When a connector is upgraded to a higher version, if there are any roles defined for any connector action in the previous version of the connector, those roles are carried forwarded to the upgraded version.
- The Playbook designer displays all the connector actions irrespective of the roles of the user creating the playbook. Also, while executing a connector action from playbook, FortiSOAR does not perform any role check. You can create a playbook as a 'Private' playbook to restrict access. For more information, see the "Playbooks Guide."
- When a connector action is being executed directly on a record, the roles of the logged in user will be matched against the roles permitted for the action.

Defining the ownership of a connector's configuration

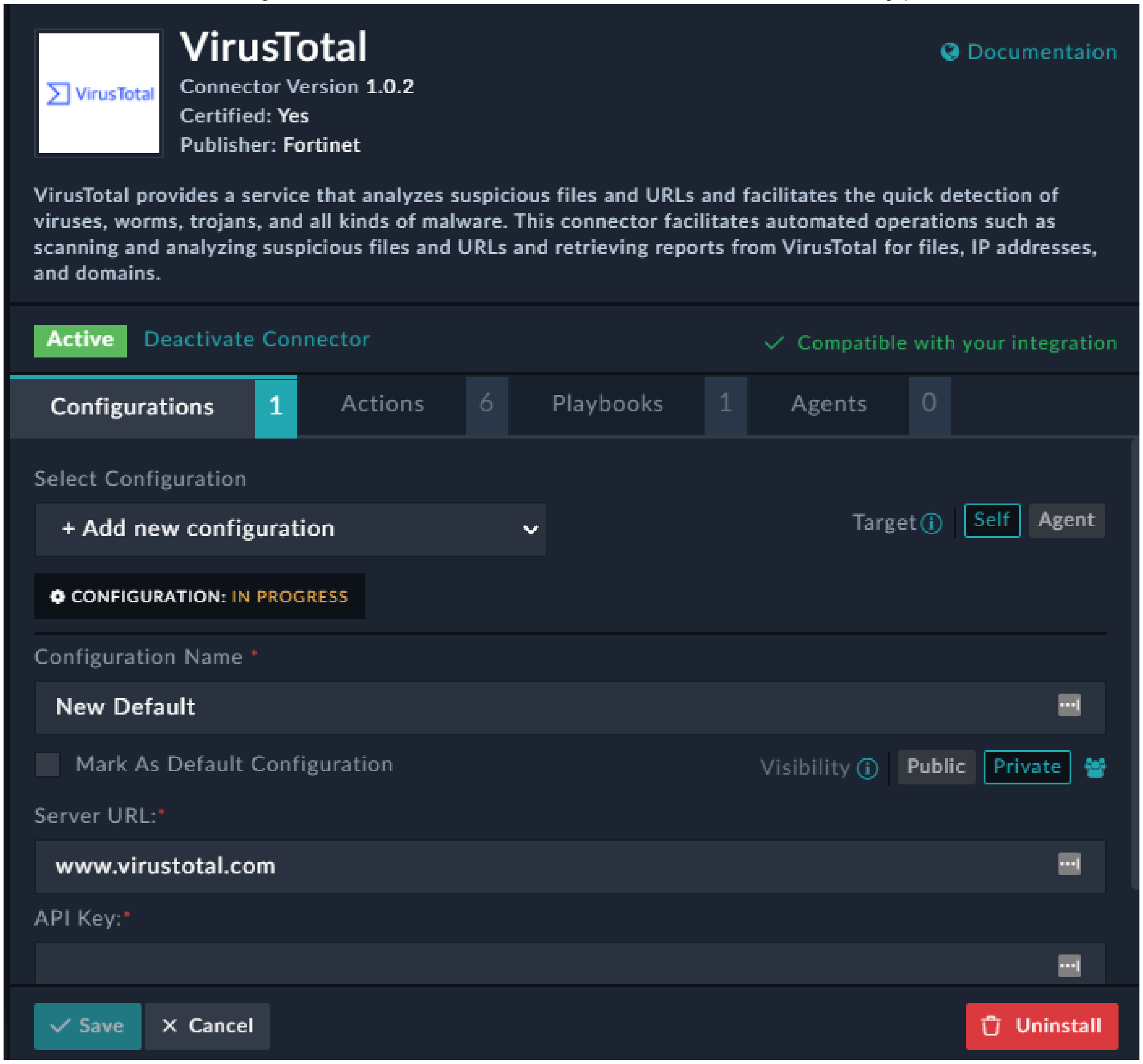
The ownership of connector configurations can also be defined by setting the visibility of a connector's configuration to 'Private'. By default, the connector configuration is set to 'Public', i.e., it can be viewed by anyone who has appropriate permissions, as defined in the Connector Store section. By setting a connector's configuration to 'Private', you are assigning ownership of that connector configuration to particular teams and thereby only allowing the assigned team owners the right to view and execute that particular connector configuration.

You must configure access control for connector configurations and actions separately for the tenant nodes. Access control for connector configurations and actions cannot be configured from the master.

To set the visibility of connector configurations to 'Private' so as to restrict the teams who can view and execute that particular connector configuration, do the following:

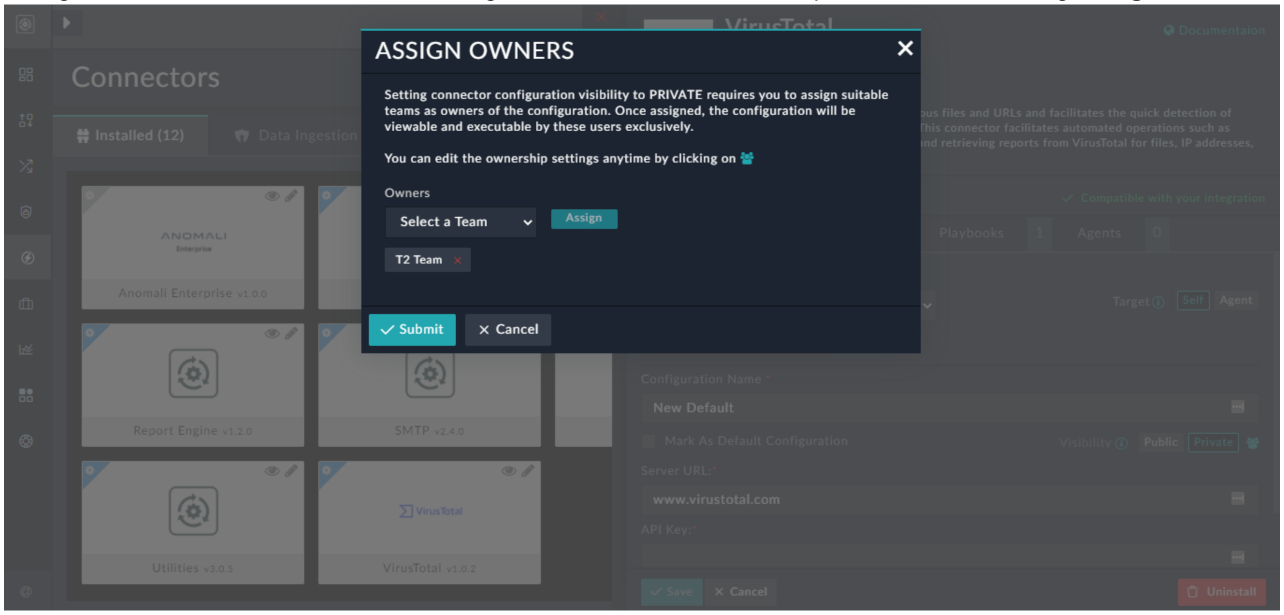
1. Log on to FortiSOAR.
2. On the left navigation pane, click **Automation** > **Connectors** > **Installed**.
3. Select the connector whose configuration you want to set to 'Private', i.e., whose configuration you want to restrict to certain teams. By default, connector configurations are set to 'Public'.

4. To mark a connector configuration as 'Private', click the **Private** button beside the **Visibility** parameter:


VirusTotal
Documentaion
Connector Version 1.0.2
Certified: Yes
Publisher: Fortinet
VirusTotal provides a service that analyzes suspicious files and URLs and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware. This connector facilitates automated operations such as scanning and analyzing suspicious files and URLs and retrieving reports from VirusTotal for files, IP addresses, and domains.
Active
Deactivate Connector
Compatible with your integration
Configurations 1
Actions 6
Playbooks 1
Agents 0
Select Configuration
+ Add new configuration
Target
Self
Agent
CONFIGURATION: IN PROGRESS
Configuration Name *
New Default
Mark As Default Configuration
Visibility
Public
Private
Server URL:*
www.virustotal.com
API Key:*
Save
Cancel
Uninstall

5. To assign the connector configuration to a particular team, once you have clicked **Private**, click the **Teams** ( ) icon to display the `Assign Owners` dialog.
6. In the `Assign Owners` dialog, you will see that the logged in user's team are assigned ownership, by default. Click the **Red Cross** beside the teams whose ownership you want to remove, and select the teams that you want to

configure as owners for this connector configuration from the **Owners** drop-down list and clicking **Assign**:


ASSIGN OWNERS

Setting connector configuration visibility to PRIVATE requires you to assign suitable teams as owners of the configuration. Once assigned, the configuration will be viewable and executable by these users exclusively.

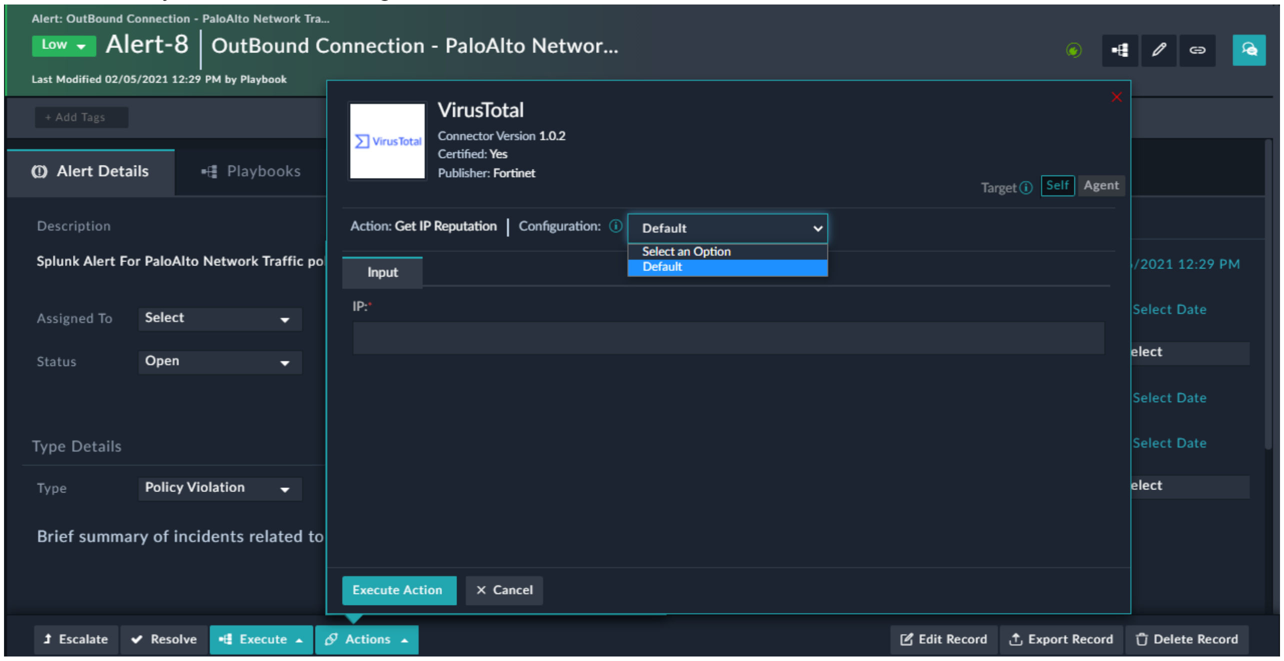
You can edit the ownership settings anytime by clicking on

Owners: Select a Team | Assign

T2 Team

Submit | Cancel

7. Click **Submit** to assign ownership of this connector configuration.
Now, in this case if there are users who are not part of the 'T2 ' team, and if they want to run the actions of the VirusTotal connector on a record for example, they will observe that the "New Default" configuration will not be visible and only the "Default" configuration is visible:


Alert-8 OutBound Connection - PaloAlto Networ...

VirusTotal
Connector Version 1.0.2
Certified: Yes
Publisher: Fortinet

Action: Get IP Reputation | Configuration: Default

Select an Option
Default

Input
IP:*

Execute Action | Cancel

Therefore, users can see only those connector configurations to which they have access, as is the case with the above example, the 'New Default' configuration is visible only to users that belong to the 'T2' team. However, users belonging to teams other than 'T2' can execute playbooks containing a connector step with the 'New Default' configuration that has been created by 'T2' users.

Building your own connector

You can write a custom connector that can retrieve data from custom sources. You can then use the connector either standalone or within FortiSOAR playbooks and perform automated operations. From version 7.0.0, you can create your own connector using the Connector Wizard. You can also write a custom connector manually, or you can use the FortiSOAR Connector SDK to develop your custom connector.

The process of installing, configuring, and using FortiSOAR-provided connectors is defined in the respective connector-specific documentation.

Permissions required for building a connector

- To create and publish a custom connector, you must be assigned a role that has a minimum of `Create`, `Read` and `Update` access on the `Connectors` module, and `Read` access on the `Application` module.
- To delete a custom connector, you must be assigned a role that has a minimum of `Create`, `Read`, `Update`, and `Delete` access on the `Connectors` module, and `Read` access on the `Application` module.

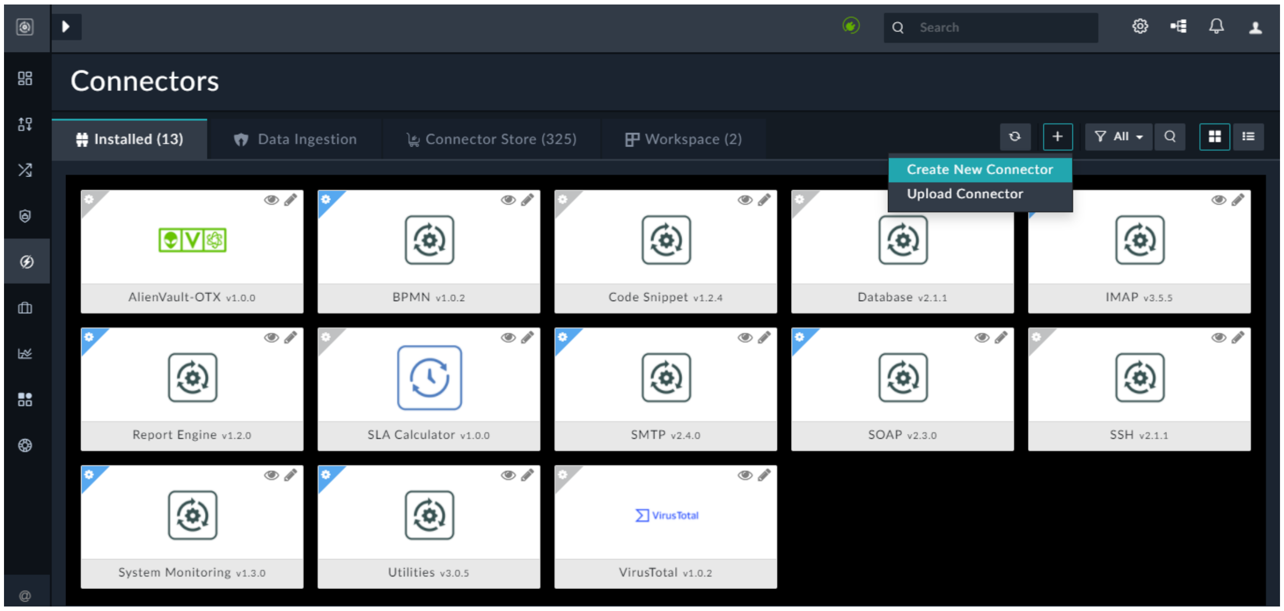
Building a connector using the Connector Wizard

From version 7.0.0, FortiSOAR provides you with a connector wizard using which you can create a new connector. You can also edit an existing connector according to your requirements by clicking on the **Edit Connector** icon on a connector card.

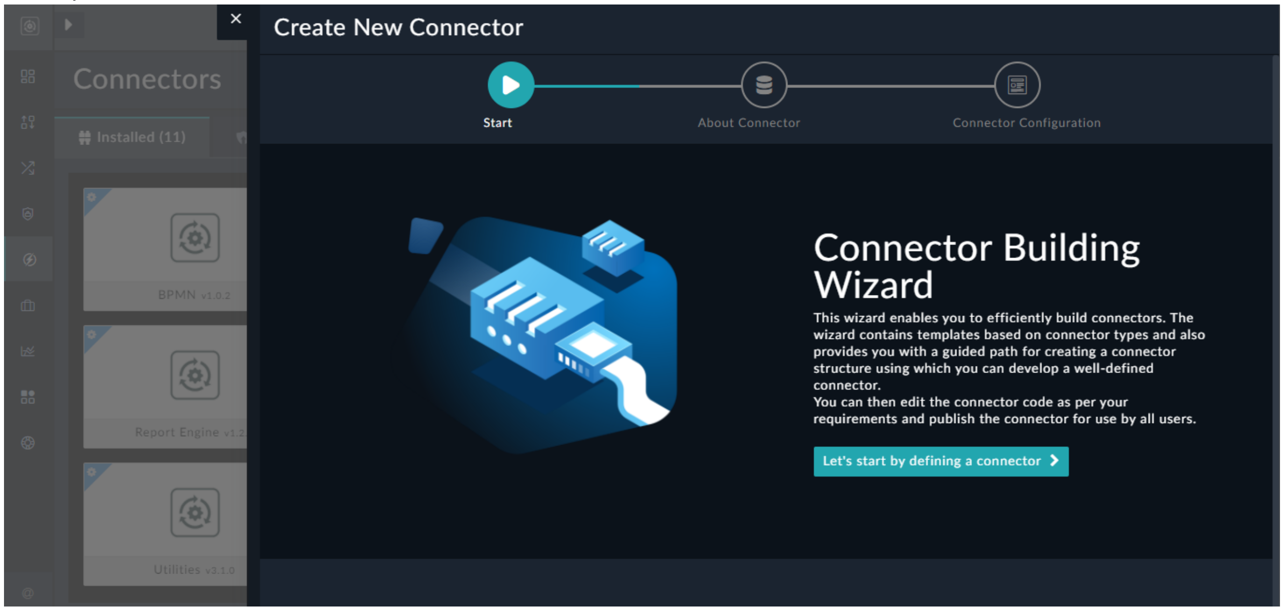
To create a new connector, do the following:

1. Log on to FortiSOAR.
2. On the left navigation pane, click **Automation** > **Connectors** > **Installed**.

3. To create a new connector, click **+** > **Create New Connector**;



This opens the "Connector Wizard":

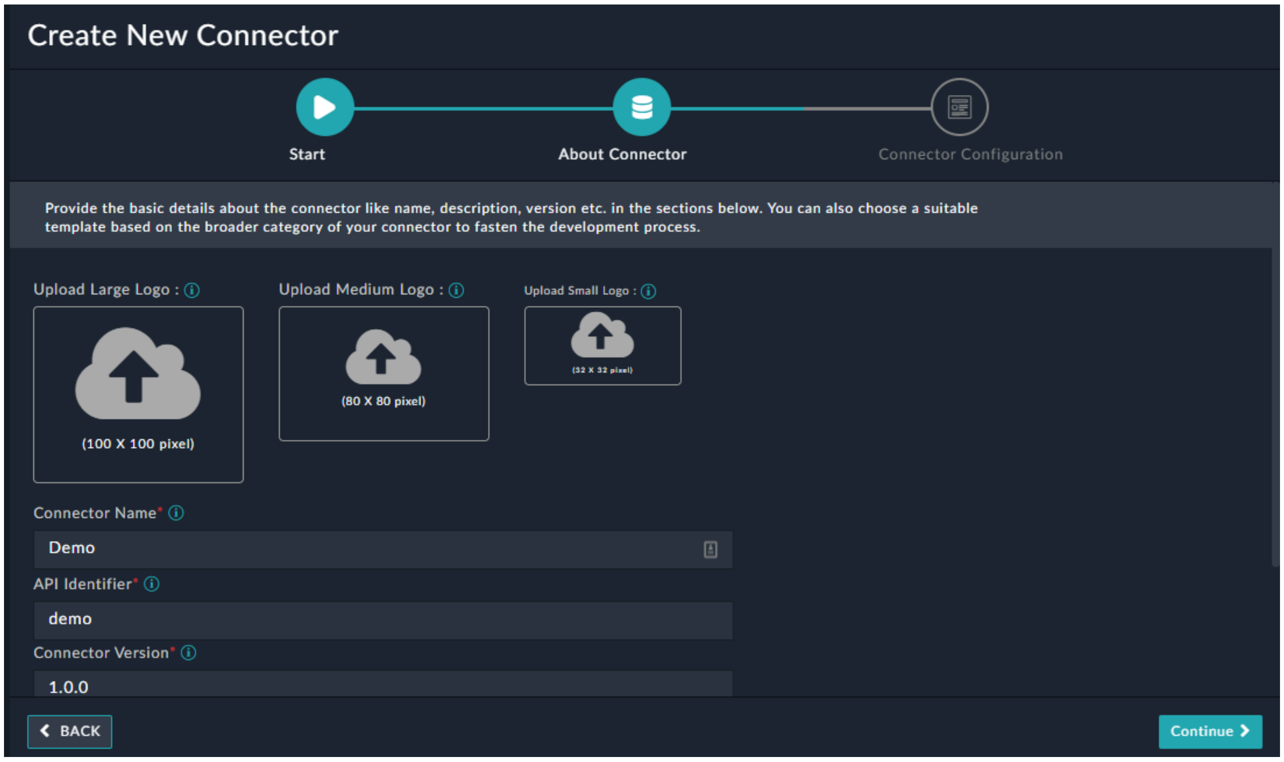


Connector Building Wizard

This wizard enables you to efficiently build connectors. The wizard contains templates based on connector types and also provides you with a guided path for creating a connector structure using which you can develop a well-defined connector.
You can then edit the connector code as per your requirements and publish the connector for use by all users.

You can also open the Connector Wizard by clicking the **Create Connector** button in the **Workspace** tab.
Click the **Lets's start by defining a connector** button to start building your connector.

4. In the Connector Wizard, in the **About Connector** screen, you need to provide the basic details such as name, description, etc. of the connector that you want to create, as well as, upload a logo for the connector, and choose a suitable template for the connector:



Create New Connector

Provide the basic details about the connector like name, description, version etc. in the sections below. You can also choose a suitable template based on the broader category of your connector to fasten the development process.

Connector Name*: Demo

API Identifier*: demo

Connector Version*: 1.0.0

You need to add the following details on this screen:

a. Click the **Upload** icons buttons to upload large, medium, or small logos for your connector. Clicking the upload button opens the **Upload Connector Logo** dialog, where you can drag-and-drop your logo, or browse to the logo on your disk and import the logo.

b. In the **Connector Name** field, add the name of the connector that you want to create.
It is a good practice to include both the name of the organization and the name of the product in the connector name, for example, Fortinet FortiSOAR.

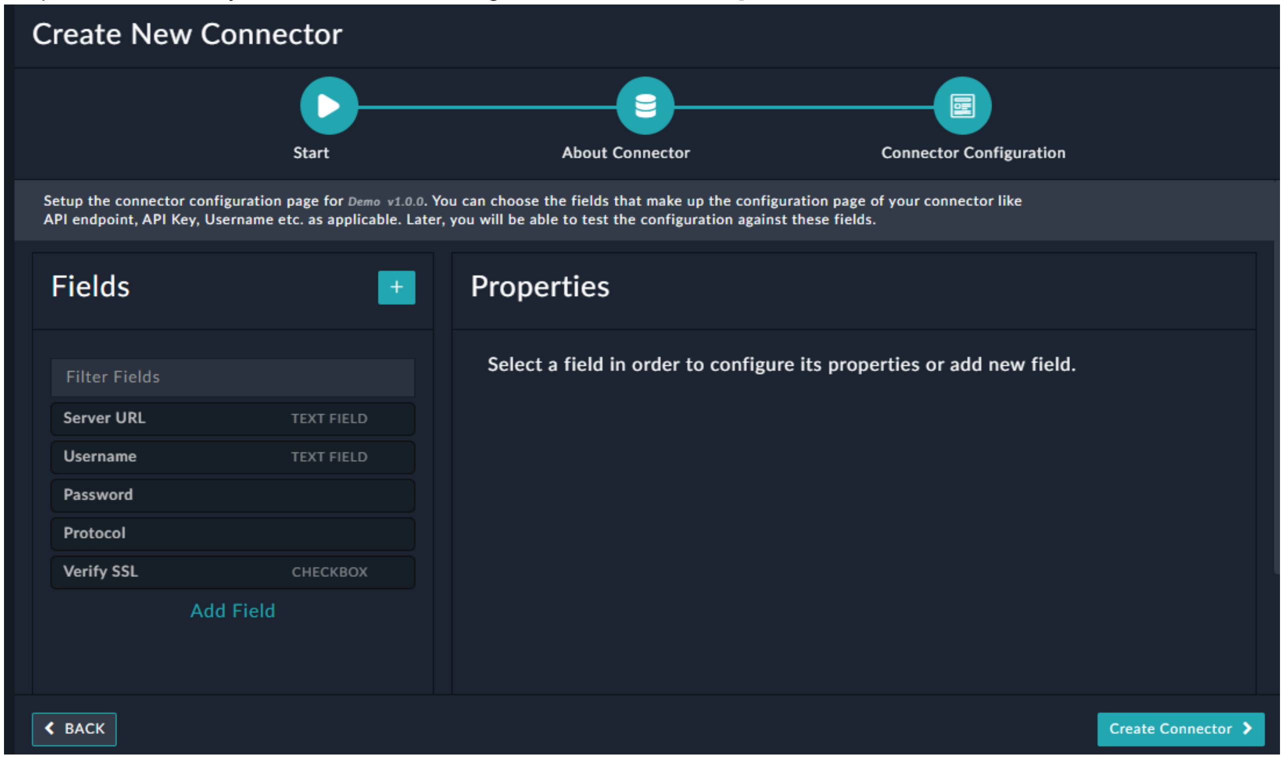
c. In the **API Identifier** field, enter a name that would be used to as a variable in the connector code to reference this connector. The variable that you use here can be alphanumeric; however, it should not contain any special characters and it must not start with a number.
Also, note that the value that you enter in this field must not match the name of any other connector that is available in the connector store. For example, you cannot enter '`virustotal`' in this field, since the virustotal connector is available in the connector store.
Note: You can change the connector name and API identifier for a connector by editing the info.json of that connector.

d. In the **Connector Version** field, enter the version of the connector in the x.y.z format. For example, 1.0.0.
Note: It is recommended that you increase the version number if you are making any changes to an installed connector.

e. In the **Description** field, enter information for the connector that you are creating, which enables users to understand more about the connector.

f. From the **Select Template** drop-down list, select the template that you want to apply to the connector that you want to create.
Templates are based on the broader category of connector, for example, threat intelligence connectors, SIEM connectors, etc. Templates automatically add suggestive actions, configuration details, and other connector details based on the type of connector you want to create, which helps to speed up the development process.

g. Once you have completed entering all the details, click **Continue**, to go to `Connector Configuration` screen.

5. Use the **Connector Configuration** screen to setup the connector configuration page. If you have specified a template in the **About Connector** screen, then you will see that some suggestive configuration fields are added based on the selected template, else you will not see any configuration fields and you require to add all the required configuration fields.
If, for example, in the **About Connector** screen, the **Threat Intel Connector** is selected from the **Select Template** drop-down list, then you will see the following `Connector Configuration` screen:

Create New Connector

Setup the connector configuration page for *Demo v1.0.0*. You can choose the fields that make up the configuration page of your connector like API endpoint, API Key, Username etc. as applicable. Later, you will be able to test the configuration against these fields.

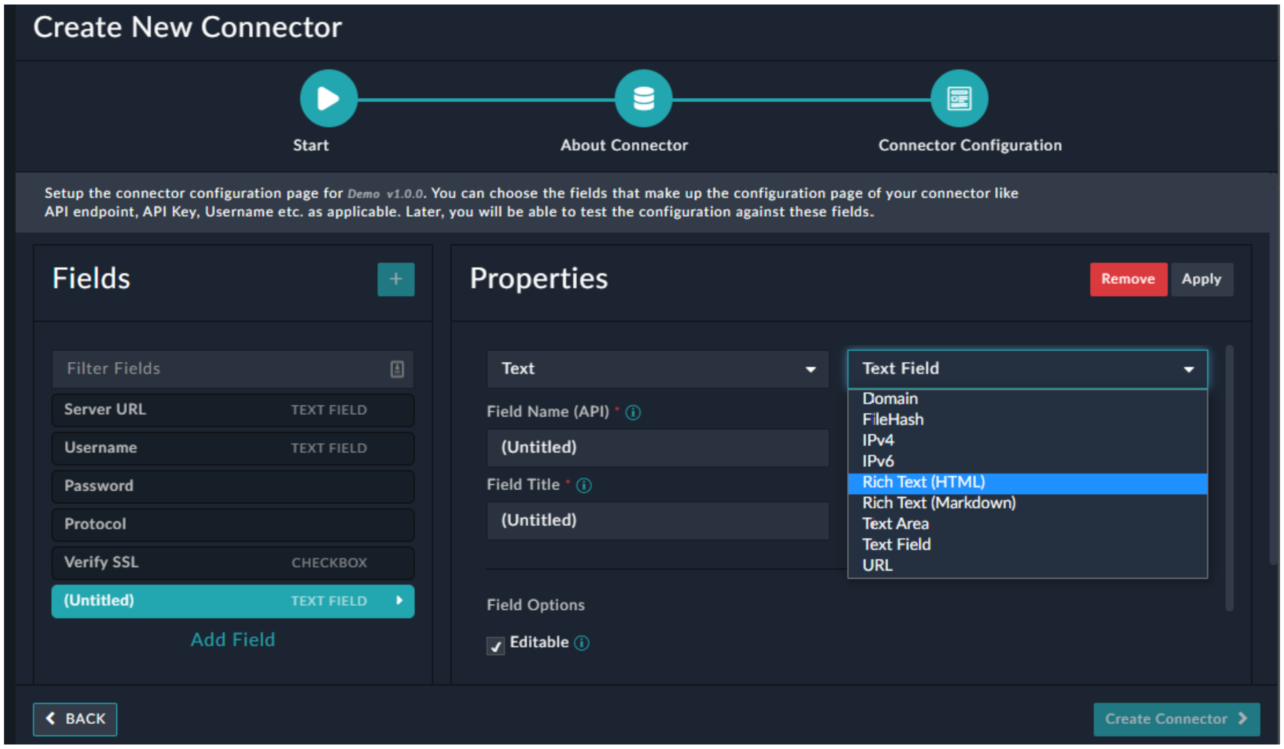
Fields

Server URL	TEXT FIELD
Username	TEXT FIELD
Password	
Protocol	
Verify SSL	CHECKBOX

Properties

Select a field in order to configure its properties or add new field.

To add fields to the connector configuration, click the **Add Field** link and can easily add properties to that field, like selecting the type of field you want to add such as Text, Integer, Email, etc., and even a sub-type of `Text` fields such as URL, Filehash, etc., all of which help you in ensuring that the user enters appropriate values in the field. You can also set other properties such as visibility and default value of the field, whether its required or not, etc.
Important: You cannot add fields that are named 'name' and 'default' to the connector configuration since, 'name' and 'default' are reserved keywords.


Create New Connector

Start · About Connector · Connector Configuration

Setup the connector configuration page for *Demo v1.0.0*. You can choose the fields that make up the configuration page of your connector like API endpoint, API Key, Username etc. as applicable. Later, you will be able to test the configuration against these fields.

Fields	
Server URL	TEXT FIELD
Username	TEXT FIELD
Password	
Protocol	
Verify SSL	CHECKBOX
(Untitled)	TEXT FIELD

Properties

Text — Text Field

Field Name (API): (Untitled)

Field Title: (Untitled)

Field Options: ✓ Editable

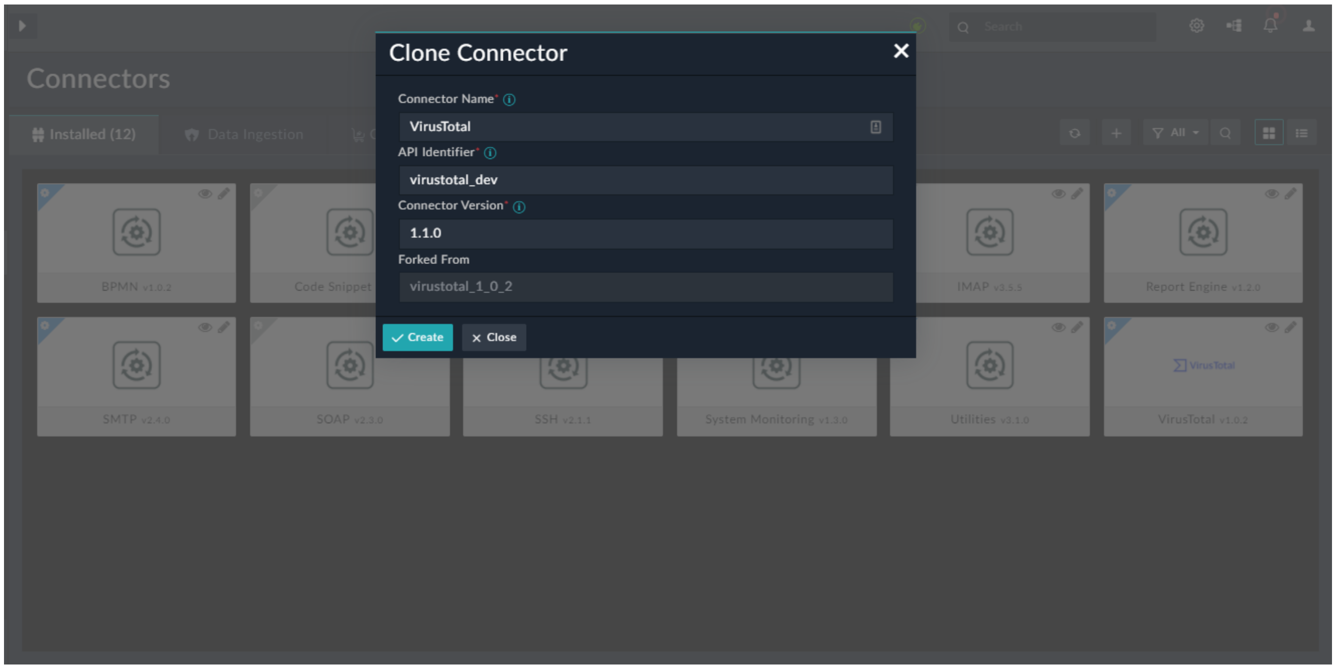
To edit existing fields and change their properties, select the field to change the properties and click **Apply**.

6. Once you have completed editing the connector configuration and to create the connector, click **Create Connector**. Once you click **Create Connector**, the code editor interface for the selected connector gets displayed for you to edit and test the connector. The created connector also gets saved in the **Workspace** tab, from where you also edit the connector. For more information, see the Editing a connector section.

Editing a connector

You can use the connector's code editor interface to edit existing connectors and build new connectors for custom use cases.

To edit an existing connector that is present in the Connector Store and which you have installed on your system, click the **Edit Connector** icon on the `Installed Connectors` page. However, before you can make any changes to a connector installed from the connector store, you are required to create a local copy of that connector. For example, click the **Edit Connector** icon on the *VirusTotal* connector. This displays a `Confirm` dialog that requiring you to clone the connector before making any changes, click **Confirm** to open the `Clone` dialog, where you should update the connector name, version, and also ensure that you have specified a unique API Identifier and then click **Create**:



Clone Connector

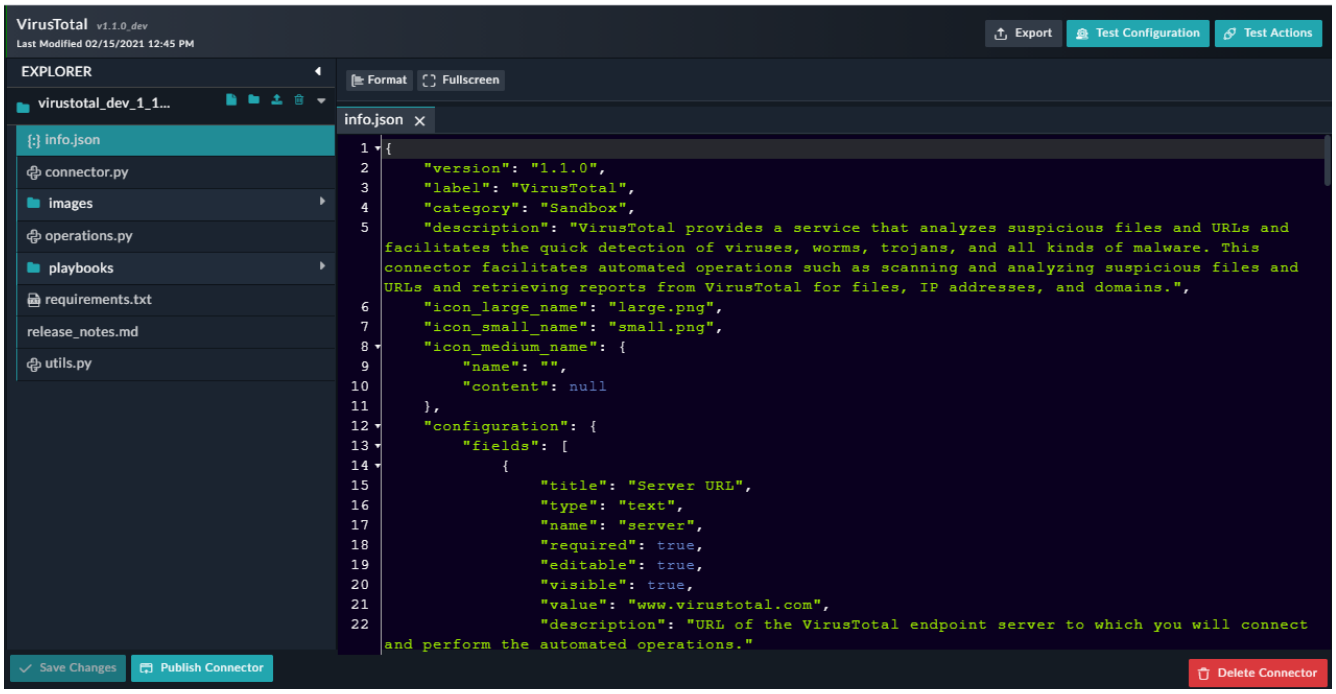
Connector Name: VirusTotal

API Identifier: virustotal_dev

Connector Version: 1.1.0

Forked From: virustotal_1_0_2

Clicking **Create** opens the code editor interface where you can make the required changes to the connector.



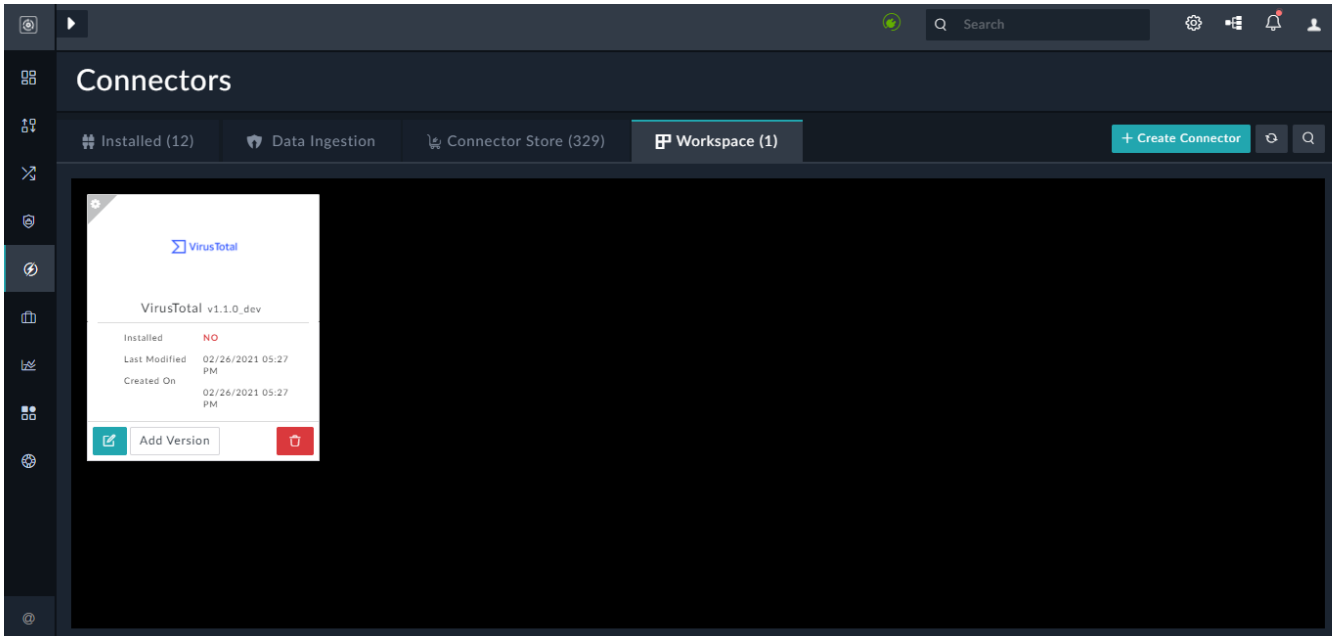
info.json

```
{
    "version": "1.1.0",
    "label": "VirusTotal",
    "category": "Sandbox",
    "description": "VirusTotal provides a service that analyzes suspicious files and URLs and
facilitates the quick detection of viruses, worms, trojans, and all kinds of malware. This
connector facilitates automated operations such as scanning and analyzing suspicious files and
URLs and retrieving reports from VirusTotal for files, IP addresses, and domains.",
    "icon_large_name": "large.png",
    "icon_small_name": "small.png",
    "icon_medium_name": {
        "name": "",
        "content": null
    },
    "configuration": {
        "fields": [
            {
                "title": "Server URL",
                "type": "text",
                "name": "server",
                "required": true,
                "editable": true,
                "visible": true,
                "value": "www.virustotal.com",
                "description": "URL of the VirusTotal endpoint server to which you will connect
and perform the automated operations."
```

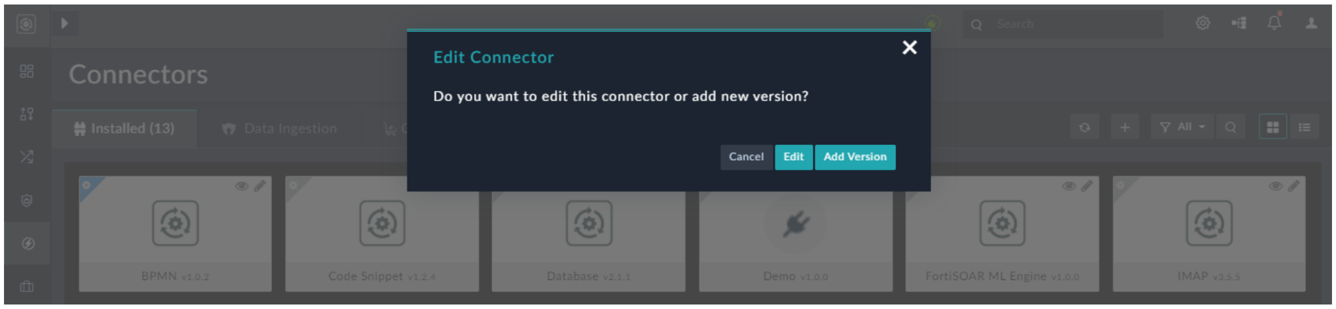
It also saves the cloned connector in the **Workspace** tab, with the name that is specified in the **API Identifier** field, i.e., '*VirusTotal v1.1.0_dev*'.

You can change the connector name and API identifier for a connector by editing the `info.json` of that connector.

To edit the connector, click the **Edit** button:



If you want to edit a custom connector that has been created using the Connector Wizard and not from the Connector Store, and which has been published (installed), you are not required to clone the connector. Instead you can choose to either edit, save, and publish the same version of the connector, or choose to add new version to the connector. For example, if you click the **Edit Connector** icon on the 'Demo' connector (created using the Connector Wizard), the Edit Connector dialog displays the **Edit** and the **Add Versions** options:



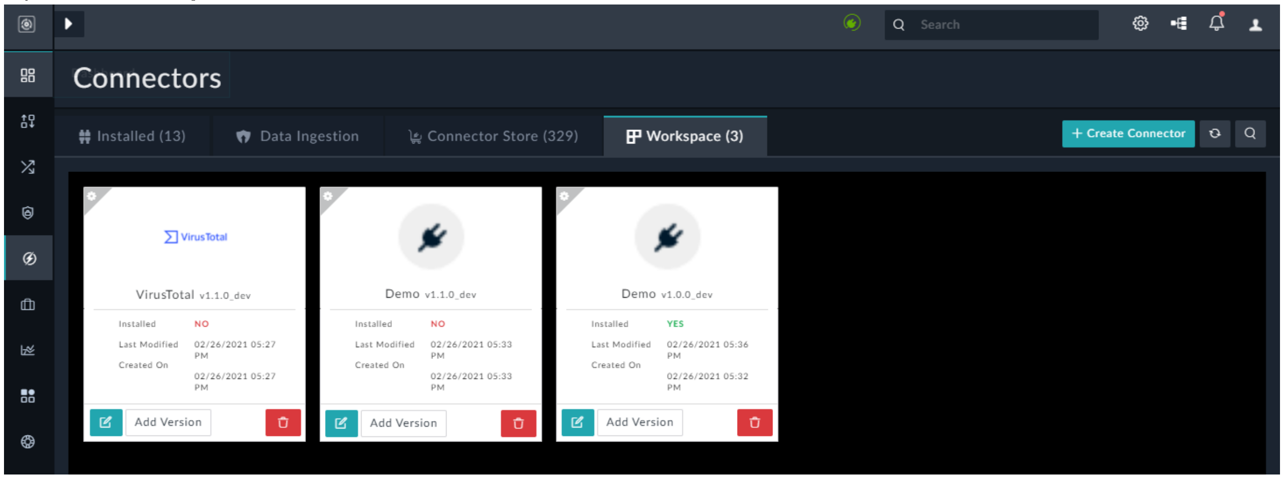
Do you want to edit this connector or add new version?

If a connector is published and is being used by other users, it is recommended that you add a version to the connector before editing the connector code, so that the changes you make do not cause any issues to existing operations done using the connector.

Similarly, once you create a connector using the "Connector Wizard", that connector is saved in the **Workspace** tab, where you can edit the connector.

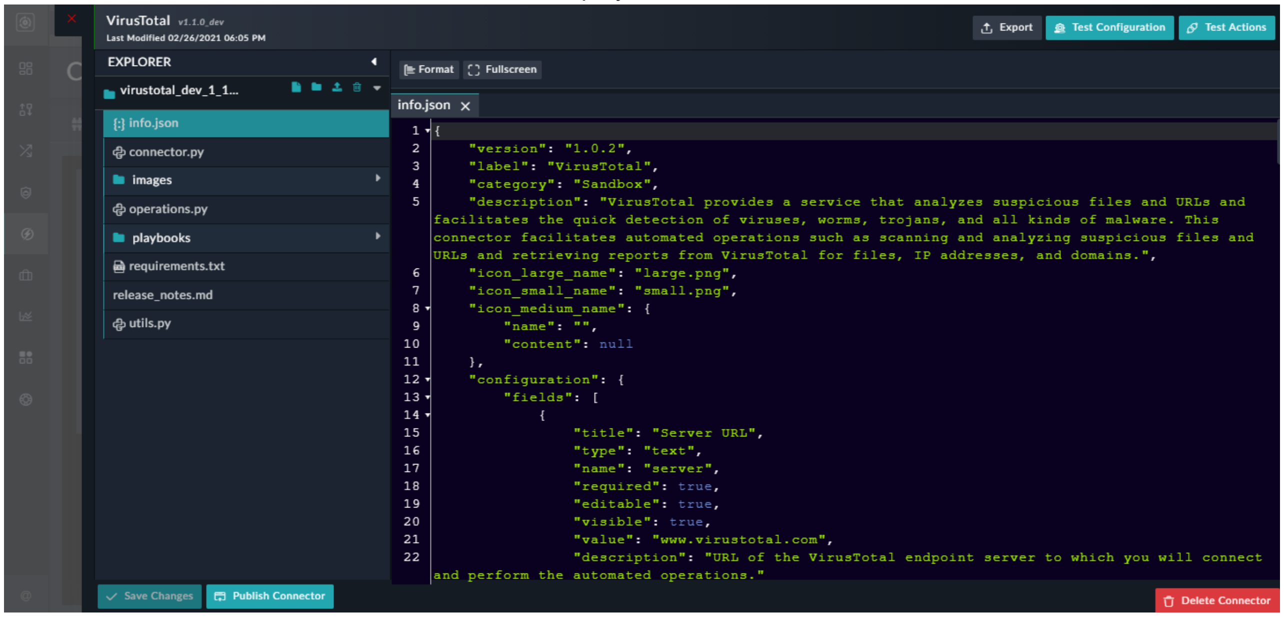
To edit a connector, do the following:

1. Open the **Workspace** tab:



VirusTotal v1.1.0_dev	Demo v1.1.0_dev	Demo v1.0.0_dev
Installed NO	Installed NO	Installed YES
Last Modified 02/26/2021 05:27 PM	Last Modified 02/26/2021 05:33 PM	Last Modified 02/26/2021 05:36 PM
Created On 02/26/2021 05:27 PM	Created On 02/26/2021 05:33 PM	Created On 02/26/2021 05:32 PM

   In the **Workspace** tab, you will see connector cards with their details such as the connector version, the created and the last modified dates, whether the connector is installed or not, etc.
   You might see more than one version of the connector such as the *Demo* connector in the above image, if you are working on multiple versions of the connector or have chosen to maintain a working copy of an installed connector in the **Workspace** tab, as is the case with *Demo v1.0.0*, i.e., in this case you have chosen the **Publish Only** option while publishing the connector.
   To add a version for the connector, click **Add Version**, which will again display the `Clone Connector` dialog.
2. To edit a connector, click **Edit Connector**, which displays the code editor interface:



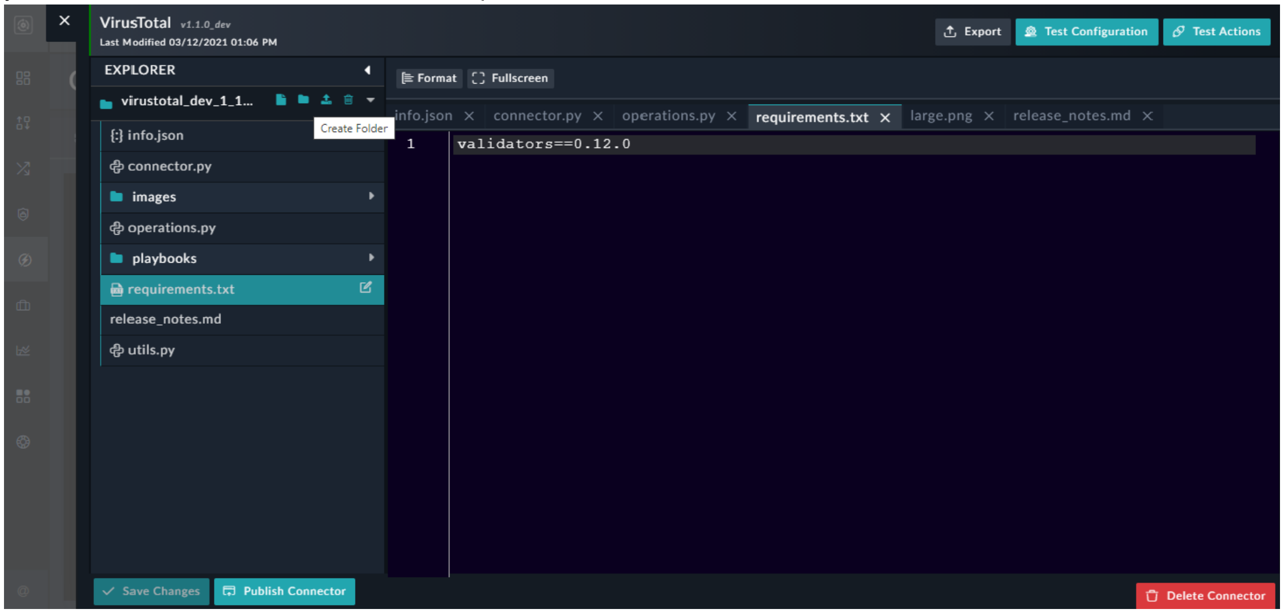
```
{
    "version": "1.0.2",
    "label": "VirusTotal",
    "category": "Sandbox",
    "description": "VirusTotal provides a service that analyzes suspicious files and URLs and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware. This connector facilitates automated operations such as scanning and analyzing suspicious files and URLs and retrieving reports from VirusTotal for files, IP addresses, and domains.",
    "icon_large_name": "large.png",
    "icon_small_name": "small.png",
    "icon_medium_name": {
        "name": "",
        "content": null
    },
    "configuration": {
        "fields": [
            {
                "title": "Server URL",
                "type": "text",
                "name": "server",
                "required": true,
                "editable": true,
                "visible": true,
                "value": "www.virustotal.com",
                "description": "URL of the VirusTotal endpoint server to which you will connect and perform the automated operations."
```

   The files and folders that are present are similar to what is described in the Writing a custom connector manually section. A brief description of these files and folders follow:
   - The `info.json` file contains information about the name and version of the connector, logo image file names, the configuration parameters, the set of functions supported by the connector, and their input parameters. The name of all the fields in the `info.json` file must be unique.
     To add an operation (Action in a connector), you must add the operation parameters in the `info.json` file, and this in turn creates a `.py` file per operation. Each operation `.py` file contains the default template format for the operation. Once the `info.json` is updated with operation it is mapped in the `builtins.py`.

- The `connector.py` file extends the base connector class and implements the `check_health` and `execute` functions.
  Note: The `connector.py` template file provides an additional function called `dev_execute` that can be used during the development phase of the connector. Code changes do not always reflect without a service restart; therefore `dev_execute` reloads the function definition at every 'execute' call to ensure that the changes made at every save in the code get immediately reflected. You can change the `execute` function in `connector.py` as follows to call the `dev_execute` function instead:
  ```
  def execute(self, config, operations, params, *args, **kwargs):
          return self.dev_execute(config, operation, params)
  ```
  However, note that reloading the function at every function call is slow and performance-intensive. So, once the development is complete and the connector is ready to use, you must revert the code to the original. For more information on `connector.py`, see the connector.py section.
- The **images** directory contains have the connector logo files.
- The **playbook** directory contains the `playbook.json` file, which contains playbook collection that you want to include with your connector.
- The `requirements.txt` file lists the additional python libraries that are required by the connector. However, the connector dependencies are not installed by default when the connector is created. If you want to test connector that is being developed, you must manually install the connector dependencies.
- The `release_notes.md` file contains information such as what's new and what's fixed, in a particular release of a connector.

Apart from the above files, it is recommended that you maintain a file (`utils.py`) that contains common functions such as datetime conversion, str to list, etc., which can be used by multiple actions in the connector.

3. Edit the existing connector as required, and click **Save Changes**.
   The connector is saved in the "Draft" state.
   The left-pane of the code editor interface contains the name of the connector along with various icons using which you create new files and folders, and also upload or delete selected files or folders:

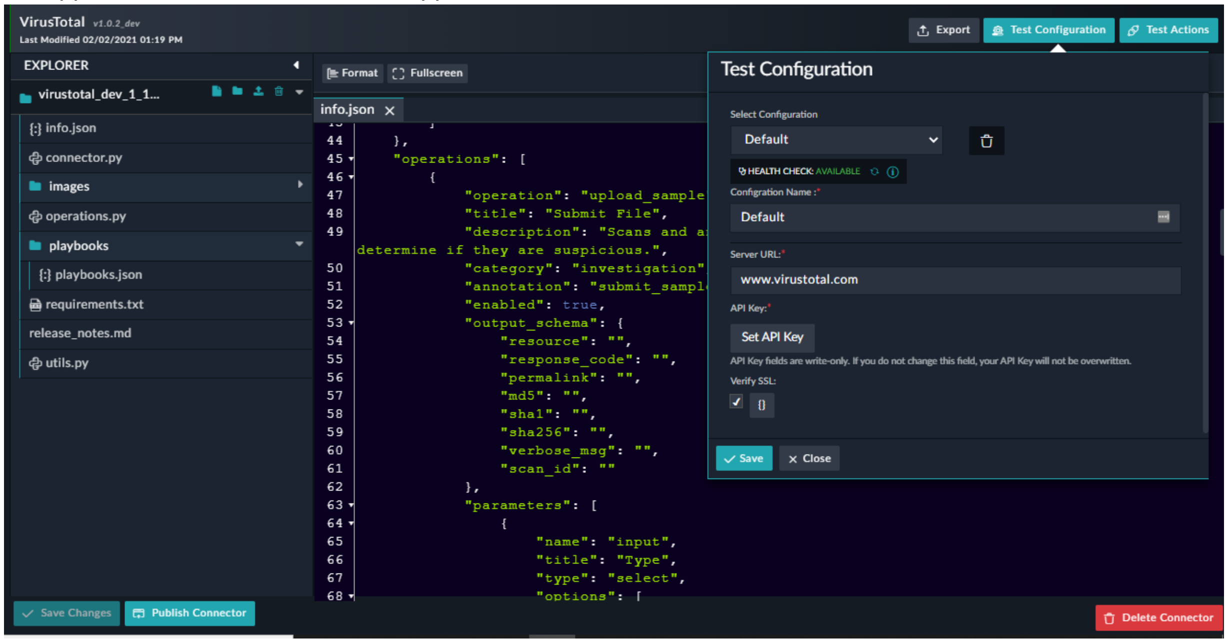


```
validators==0.12.0
```

You can also rename connector files such as the `requirements.txt`, `release_notes.md`, etc., by selecting the file row and clicking the **Edit Filename** icon, which displays a `Rename File` dialog in which you can specify the new filename and click **Save**.
You can also perform the following operations on the code editor interface:

- **Code Formatting**: To lint your code automatically and make the code more human-readable and error-free (programming and programming errors), select the entire code in the editor and click the **Format** button.

- **Full Screen**: To get a better working view and make the editor go full-screen, click the **Fullscreen** button. To exit the full screen, press `ESC`.
- **Export**: To export the connector as a `.tgz` file, click the **Export** button. You might want to import the exported connectors' `.tgz` file into another system.
- **Testing the configuration**: While developing a connector, you can perform health check operation on the connector. Clicking the **Test Configuration** button opens a dialog containing the specified configuration parameters, where you can input the values, save the configuration, and perform the health check. If the configuration is valid, i.e., the configuration parameters and values specified are correct, then the health check will appear as "Available", else it will appear as "Disconnected".



VirusTotal v1.0.2_dev
Last Modified 02/02/2021 01:19 PM

EXPLORER
virustotal_dev_1_1...
info.json
connector.py
images
operations.py
playbooks
playbooks.json
requirements.txt
release_notes.md
utils.py

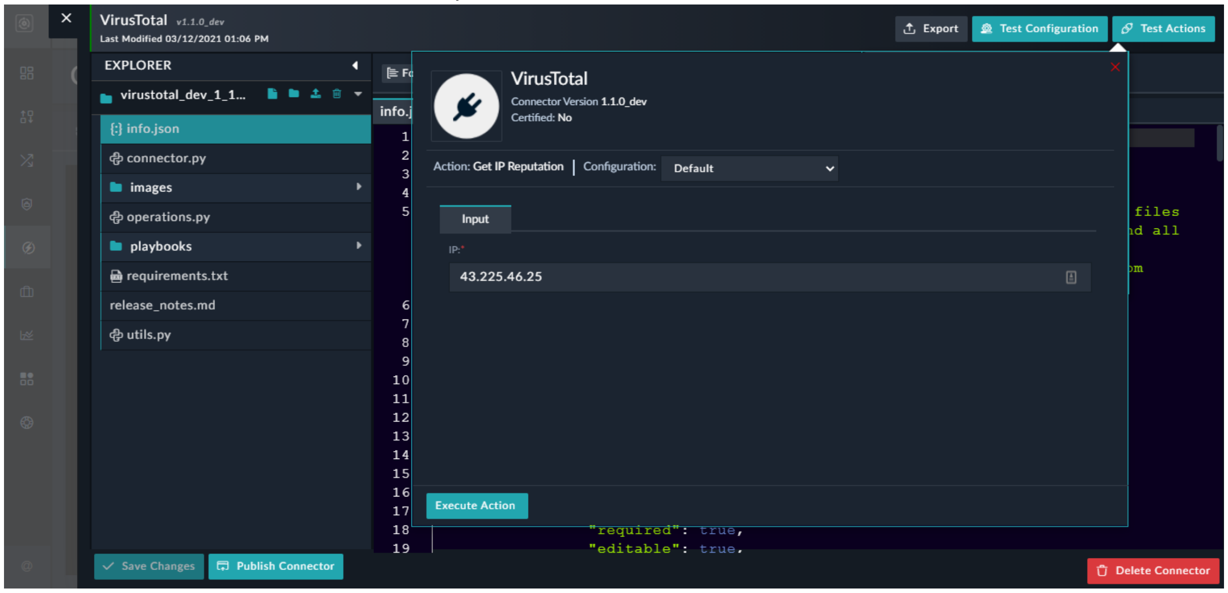
info.json

```
        },
        "operations": [
            {
                "operation": "upload_sample
                "title": "Submit File",
                "description": "Scans and a
determine if they are suspicious.",
                "category": "investigation"
                "annotation": "submit_sampl
                "enabled": true,
                "output_schema": {
                    "resource": "",
                    "response_code": "",
                    "permalink": "",
                    "md5": "",
                    "sha1": "",
                    "sha256": "",
                    "verbose_msg": "",
                    "scan_id": ""
                },
                "parameters": [
                    {
                        "name": "input",
                        "title": "Type",
                        "type": "select",
                        "options": [
```

Test Configuration

Select Configuration: Default
HEALTH CHECK: AVAILABLE
Configration Name:* Default
Server URL:* www.virustotal.com
API Key:* Set API Key
API Key fields are write-only. If you do not change this field, your API Key will not be overwritten.
Verify SSL: ✓

- **Testing the actions**: You can use the saved configuration for testing connector actions for their output/errors. Clicking the **Test Actions** button displays a list of actions for that connector. Click the action that you want to test to display a dialog containing the input parameters for that action. Enter the values for the input parameters and click **Execute Action** to view the output for that action.



VirusTotal v1.1.0_dev
Last Modified 03/12/2021 01:06 PM

EXPLORER
virustotal_dev_1_1...
info.json
connector.py
images
operations.py
playbooks
requirements.txt
release_notes.md
utils.py

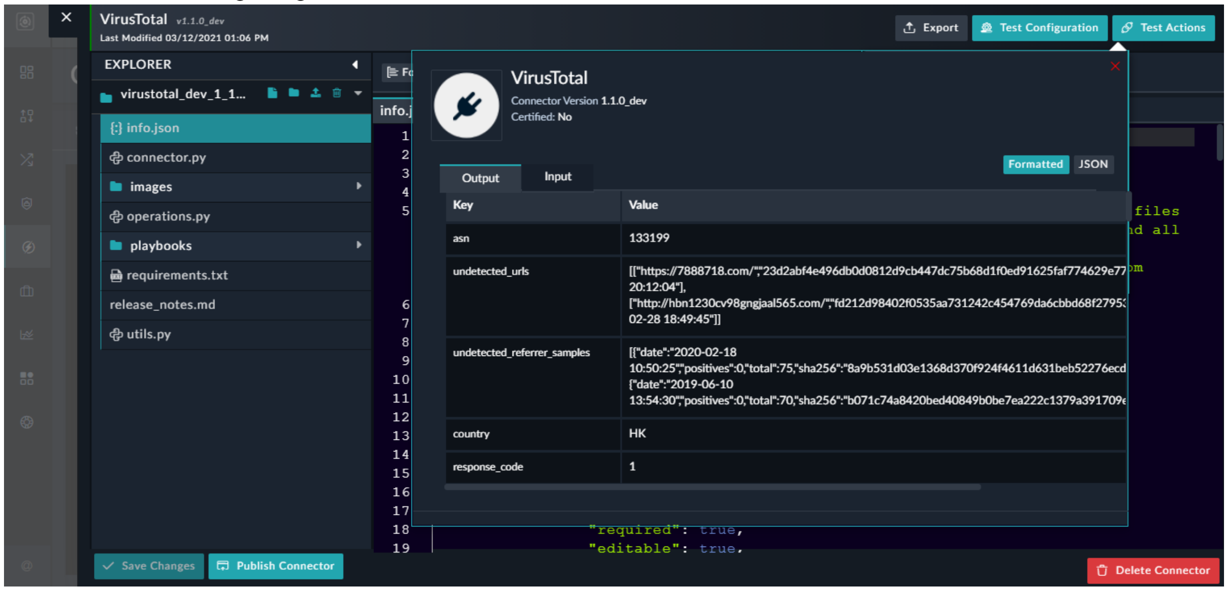
VirusTotal
Connector Version 1.1.0_dev
Certified: No

Action: Get IP Reputation | Configuration: Default

Input
IP:* 43.225.46.25

```
                "required": true,
                "editable": true,
```

Once the connector action is executed, you can see the formatted output of the action, in a tabular format, as shown in the following image:



VirusTotal
Connector Version 1.1.0_dev
Certified: No

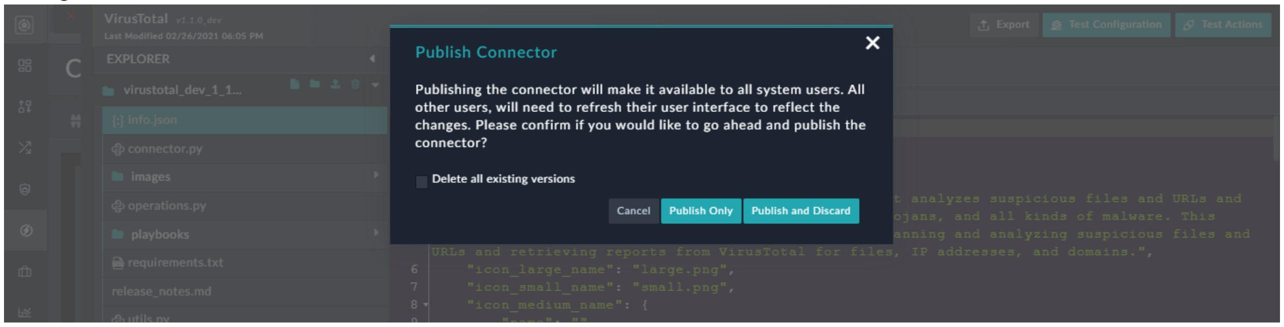
Output | Input | Formatted | JSON

Key	Value
asn	133199
undetected_urls	[["https://7888718.com/","23d2abf4e496db0d0812d9cb447dc75b68d1f0ed91625faf774629e77 20:12:04"], ["http://hbn1230cv98gngjaal565.com/","fd212d98402f0535aa731242c454769da6cbbd68f2795 02-28 18:49:45"]]
undetected_referrer_samples	[{"date":"2020-02-18 10:50:25","positives":0,"total":75,"sha256":"8a9b531d03e1368d370f924f4611d631beb52276ecd {"date":"2019-06-10 13:54:30","positives":0,"total":70,"sha256":"b071c74a8420bed40849b0be7ea222c1379a391709e
country	HK
response_code	1

```
"required": true,
"editable": true,
```

- **Deleting the Connector**: To delete the connector, click the **Delete Connector**. button.

4. Once you have completed making the changes, click **Publish Connector** to display the `Publish Connector` dialog:



Publish Connector

Publishing the connector will make it available to all system users. All other users, will need to refresh their user interface to reflect the changes. Please confirm if you would like to go ahead and publish the connector?

Delete all existing versions

Cancel | Publish Only | Publish and Discard

```
t analyzes suspicious files and URLs and
ojans, and all kinds of malware. This
anning and analyzing suspicious files and
URLs and retrieving reports from VirusTotal for files, IP addresses, and domains.",
    "icon_large_name": "large.png",
    "icon_small_name": "small.png",
    "icon_medium_name": {
```

To delete all existing versions of the connector from your system, click the **Delete all existing versions** checkbox. The publish operation has the following options:

- **Publish Only**: This publishes the connector and makes it available on the **Installed** tab for all the users of the system and use it in playbooks or run directly on records. However, a working copy for the same is also maintained in the **Workspace** tab.
- **Publish and Discard**: This publishes the connector, makes it available on the **Installed** tab for all the users of the system, and also removes this connector from the **Workspace** tab.

Removing images of a connector installed from the connector store

If you are editing a connector that is part of the connector store, for example 'VirusTotal' and you want to remove the images of such a connector, do the following:

1. Open the connector in the code editor interface, and click the **images** folder.
2. Select the image that you want to delete and then click the **Delete** icon, then click **Delete** on the confirmation dialog.
3. In the code editor interface, click **info.json** and edit the `info.json` file to remove the file references of the deleted image. For example, if you have deleted the large logo for the connector (`large.png`), then in the `info.json` file,

remove its reference:
`"icon_large_name": "large.png"`

4. To save your changes, click **Save Changes**.

Debugging connectors

- You can add logger statements to test your connector's code paths. Use the `get_logger` utility function to initialize the logger. You can import `get_logger` from `connectors.core.connector`, and then declare the logger as `logger = get_logger('<connector name>')`. Then you can add `logger.error('<logger_message>')` to the `operation.py` file that you want to debug.
- If your connector's configuration or action fails, you can check the connector logs. All the connector logs are written to the `/var/log/cyops/cyops-integrations/connectors.log` file and follow the format: `%(asctime)s %(levelname)s %(name)s %(module)s %(funcName)s(): %(message)s`
  The default log level is `WARN`, however, you can change it to `INFO` or `DEBUG`.
  For example, to change the log level to `INFO`:
  Open the `/opt/cyops-integrations/integrations/configs/config.ini` file and set `connector_logger_level = 'INFO'`, and then restart the `uwsgi` service.

Writing a custom connector manually

Perform the following steps to write a custom connector manually:

1. Create a Connector Template Directory Structure with the required files.
2. Import the connector into FortiSOAR.
3. Check the health of the connector.
4. Add a connector operation to a playbook.

Connector Template Directory Structure

Create the following folder structure, with a folder with the connector name at the top and files within it:

connectorname folder

```
--+ playbooks
---+ __init__.py
---+ playbooks.json
--+ connector.py
--+ info.json
--+ images
--+ requirements.txt
--+ packages
---+ <package_name>
```

Python 3 is required for developing your connectors.

info.json

The `info.json` file contains information about the name and version of the connector, logo image file names, the configuration parameters, the set of functions supported by the connector, and their input parameters. The name of all the fields in the info.json file must be unique.

Ensure that the name of the connector in the `info.json` and the name of the connector folder matches exactly.

You can configure the following parameters, fields, and operations while writing connectors: description of connector and actions, tooltip, placeholder, conditional fields, backward compatibility (using visible_onchange parameter), validation using regex, label, apioperations, and grouping of fields. These parameters are explained in the Notes following the sample info.json file.

Following is an example of an `info.json` file:

```
{
 "name": "sampleConnector",
 "label": "sampleConnector",
 "description": "sampleConnector connector description",
 "publisher": "publisherName",
 "cs_approved": true/false,
 "cs_compatible": true/false,
 "version": "1.0.0",
 "category": "categoryType",
 "help_online": "link to online documentation",
 "icon_small_name":"small_icon.jpeg",
 "icon_large_name":"large_icon.jpeg",
 "configuration": {
   "fields": [
     {
       "title": "fieldname",
       "required": true/false,
       "editable": true/false,
       "visible": true/false,
       "type": "text",
       "description": "text",
       "name": "user",
       "tooltip": "text for the tooltip",
       "placeholder": "placeholder text",
       "validation":{
            "pattern":"regex pattern for validation",
            "patternError":"text for the error message if validation fails"
          }
     }]
 },
 "operations": [
   {
     "operation": "function_template",
     "title": "Sample Function",
     "description": "description of the operation",
     "category": "categoryType",
     "annotation": "sample_annotation",
        "parameters": [{
          "title": "Sample Input1",
          "required": true/false,
          "editable": true/false,
          "visible": true/false,
          "type": "text",
          "name": "input1",
          "description": "text",
          "value": "default value1"
          "tooltip": "text",
        },
        {
          "title": "Sample Input2",
          "required": true/false,
          "editable": true/false,
          "visible": true/false,
          "type": "text",
          "name": "input2",
          "description": "text",
          "tooltip": "text",
          "value": "",
          "options": ["A", "B"],
          "onchange": {
               "A": [{
                 "title": "User2",
                 "required": true/false,
                 "editable": true/false,
                 "visible": true/false,
                 "visible_onchange":false,
                 "type": "text",
                 "name": "user2",
                 "description": "text",
                            "tooltip": "text",
                 "value": "admin1"
               }],
               "B": [{
                 "title": "User2",
                 "required": true/false,
                 "editable": true/false,
                 "visible": true/false,
                 "type": "integer",
                 "name": "user2",
                 "description": "text",
                            "tooltip": "text",
                 "value": 12345
               },{
                 "title": "Comment1",
                 "required": true/false,
                 "editable": true/false,
                 "visible": true/false,
                 "type": "text",
                 "name": "comment1",
                 "description": "text",
                            "tooltip": "text",
                 "placeholder": "Add comment1",
                 "value": "",
                 "onchange": {
                   "placeholder attribute": [{
                     "title": "placehoder attribute title",
                     "required": true/false,
                     "editable": true/false,
                     "visible": true/false,
                     "type": "checkbox",
                     "name": "placehoder attribute name",
                     "description": "text",
                                "tooltip": "text",
                     "value": false
                   }
                 ]}
               }]
          }
        }],
        "enabled": true,
        "output_schema": {"key1": "", "key2": []}
      },
      {
               "operation": "Sample Operation 2",
               "category": "containment",
               "annotation": "sample_op",
               "title": "title of the operation",
               "description": "description of the operation",
               "is_config_required": true/false,
               "parameters": [
                   {
                       "title": "title of list",
                       "type": "text",
                       "name": "sample_list",
                       "required": true/false,
                       "editable": true/false,
                       "visible": true/false,
                       "description": "description of the operation",
                       "tooltip": "text"
          },
                   {

                       "type": "label",
                       "label": "label text",
                       "visible": true
          },
                   {
                       "title": "title of list",
                       "type": "text",
                       "name": "sample_list",
                       "description": "description of the operation",
                       "required": true/false,
                       "editable": true/false,
                       "visible": true/false,
                       "class": "group-element",
                       "tooltip": "text"
          },
                   {
                       "title": "title of list",
                       "type": "text",
                       "name": "sample_list",
                       "description": "description of the operation",
                       "required": true/false,
                       "editable": true/false,
                       "visible": true/false,
                       "class": "group-element",
                       "tooltip": "text"
          },
                   {
                       "title": "title of list",
                       "type": "text",
                       "name": "sample_list",
                       "description": "description of the operation",
                       "required": true/false,
                       "editable": true/false,
                       "visible": true/false,
                       "class": "group-element last",
                       "tooltip": "text"
          }
         ],
               "enabled": true,
               "output_schema": {}
       }
    ],
    "playbooks": [
      {
      }
    ]
   }
```

Notes:

- The version of the connector must be in the **x.y.z** format, for example, **1.0.0**. Version must consist of valid integers, for example, "1.15.125" is a valid version.
- The `output_schema` defines the keys that are present in the output json on the execution of an operation. The info.json contains some common keys. However, the output json can have additional keys based on the input parameters. You can use these json keys to set the input for subsequent Playbook Steps, using the Dynamic Values. For more information, see the *Dynamic Values* section in "Playbooks."
- If you want to add online and/or offline documentation for your connector, add the following to your info.json:
  ```
  help_file: "name of pdf file in the connector folder"
  help_online: "link to the online documentation for the connector"
  ```
- Input types supported: text, checkbox, integer, decimal, datetime, phone, email, file, richtext, json, textarea, image, select, and multiselect.
  For `select` and `multiselect`, you can provide the list of inputs using the `options` key.
  For example,
  ```
  {
        "title": "Sample Field",
        "name": "sample",
        "required": "true",
        "editable": "true",
        "visible": "true",
        "type": "select",
        "options": ["A","B","C"]
  }
  ```

- `category` and `annotation` within *operations*: The category defines the category for the connector that you are adding, and it must be one of the following: investigation, remediation, containment and miscellaneous.
  The annotation defines the operation or function that will be performed. An annotation is unique and belongs to only one category, i.e., you must not add an annotation to multiple categories.
  If you do not define any category in the `info.json` file, then by default, the annotation is added to the miscellaneous category.
  The category name must contain only lower-case alphabets. The annotation name must contain only lower-case alphabets, underscores, and numbers.
  Category and annotations must always come together.
  Multiple operations within a connector can use the same annotation.
- `description`: You can add a description for the connector, which will be visible on the connector page in FortiSOAR and you can also add a description for the action or operation, which will be visible on the connector step page in the Playbook Designer.
- `is_config_required`: By default, the `is_config_required` key is set to `true` (default), which means that the connector uses the configuration specified in the connector configuration to execute an action. If you set the `is_config_required` key to `false`, then configurations are not required to execute an action and in this case, you can use dynamic fields specify the configurations.
  Note: This key is applicable at the action level, i.e., limited to a single action.
- `tooltip`: You can add the tooltip parameter to any field to display information about that particular field. Note that if you do not want a tooltip against any field, then you must remove the tooltip parameter from that field in the `info.json`. You must not pass `""` or `null` as values to the tooltip parameter.
- `placeholder`: You can add the placeholder parameter for text and select fields, which will display placeholders for those fields on the connector step page in the Playbook Designer.
- conditional fields (`onchange`): You can add the `onchange` parameter to fields, which you can use to display other fields or subfields conditionally based on the user input for that field. For example:

```
{
  "options": ["Now", "Yesterday"],
  "onchange": {
              "Now": [{
                "title": "Comment3",
                "required": true,
                "editable": true,
                "visible": true,
                "type": "text",
                "name": "comment3",
                "value": ""
              }],
              "Yesterday": [{
                "title": "Comment4",
                "required": true,
                "editable": true,
                "visible": true,
                "type": "text",
                "name": "comment4",
                "value": ""
              }]
  }
}
```

- `visible_onchange`: For backward compatibility of the connector you can use the `visible_onchange` parameter for conditional fields (`onchange` parameter).
  If you set `visible_onchange` to `false`, then this field will be hidden in FortiSOAR UI, and if you set `visible_onchange` to `true`, or if it is not present for any field, then this field is visible in FortiSOAR UI.

- `validation`: You can add regex validation to `text` and `textarea` fields. You can also add the error that will be displayed in case the validation fails. For example:

```
{
 "validation":{
            "pattern":"\\d+",
            "patternError":"this is not a number"
          }
}
```

- `label`: You can add the label field to display text on the connector UI. For example:

```
{
"type": "label",
"label": "this is my label",
"visible": true
}
```

- `grouping`: You can group fields based on your categorization using `"class": "group-element"`. For the last field in the group, use `"class": "group-element last"`.
  For example:

```
{
       "operation": "block_applications",
       "parameters": [
          {

              "type": "label",
              "label": "this is my label",
              "visible": true
       },
           {
               "title": "Applications Names(List Format)",
                "type": "text",
                "name": "app_list",
                "required": true,
                "editable": true,
                "visible": true,
                "class": "group-element",
                "tooltip": "Block Applications Names (List Format)"
       },
           {
               "title": "Applications Names(List Format)",
               "type": "text",
                "name": "app_list",
                "required": true,
                "editable": true,
                "visible": true,
               "class": "group-element last",
              "tooltip": "Block Applications Names (List Format)"
       }
     ]
 }
```

- `apiOperations`: You can fetch options for the `select` and `multiselect` fields from the API that you have defined in your operation. The parameters to this operation will be what users have entered in the configuration of this operation and the target field. apiOperations is not supported for the connector configuration.

Note: To hide this operation in the from the **Actions** drop-down list in the Playbook Designer set `"visible": false` for this operation.

- `supports_remote`: Some connector actions cannot work when run on agents in a segmented network. Therefore, parameters of those actions must be marked as `"supports_remote": False`. For example:

```
"title": "Enable abc service",
"name": "abc_service",
"supports_remote":  False
```

- `conditional_output_schema`: Support for multi-option or dynamic output schema in connectors. In every operation that requires the dynamic output schema, you must add `conditional_output_schema` as a *key* in an array of objects. Each object will contain the condition, which you should add within `{{ }}` and then you must add the corresponding output schema for each condition. It is recommended that you add a default schema, using `"condition": "{{true}}"`, at the end of the defined conditions. The default schema will be used if none of the defined conditions are met.
  Notes:
  a. If you are using multiple conditions for evaluation such as a combination of `&&` and `||`, then you must wrap the whole condition within `()`.
     For example, `{{(command === 'ls' || command === 'ssh' )}}`
     `{{(command === 'ls' && port === 5895 )}}`
  b. If you are using the condition with the checkbox field, add the condition as:
     `{{ checkbox === true}}`
  c. If you are using the condition with the multi-select field, add the condition as:
     `{{multsel.toString() === (['option1','option2']).toString()}}`
  d. If you are using the condition with the `json` field, add the condition as:
     `{{jsonfieldname === '{\"key\":\"value\"}'}}`
     For example, `{{jsonfieldname === '{\"name\":\"fortinet\"}'}}`
     Important: In order to support versions earlier to the 4.12.0 version, `output_schema` will also always be present.
     Example of a `conditional_output_schema` definition:

```
{
       "conditional_output_schema": [
          {
                 "condition": "{{command === 'ls'}}",
                 "output_schema": {
                     "op_result": "",
                     "op_status": ""
                 }
             },
             {
                 "condition": "{{command === 'ssh'}}",
                 "output_schema": {
                     "result": "",
                     "status": ""
                 }
             },
             {
                 "condition": "{{true}}",
                 "output_schema": {
                     "result_default": "",
                     "status": ""
                 }
             }
```

```
            ]
        }
```

connector.py

The `connector.py` class extends the base connector class and implements the `check_health` and `execute` functions. Following is a skeleton of this class:

```
from connectors.core.connector import Connector, get_logger
logger = get_logger('<connector_name>')

class Template(Connector):
   def execute(self, config, operation, params, **kwargs):
       supported_operations = {'operation_1': function_template}
       return supported_operations[operation](config, params)

   def check_health(self, input):
       return True
```

Notes:

- All imports for the connector files should be relative. For example, if your `util.py` is parallel to `connector.py` and you want to import `util.py` then you must import it as: `import .util`.
- `get_logger` is a utility function to initialize the logger. You can import `get_logger` from `connectors.core.connector`, and then declare the logger as `logger = get_logger('<connector name>')`. All the connector logs are written to the `/var/log/cyops/cyops-integrations/connectors.log` file and follow the format: `%(asctime)s %(levelname)s %(name)s %(module)s %(funcName)s(): %(message)s`.
- In addition to the `execute` and `check_health` functions, the following optional and additional functions are also available:
  `on_add_config(self, config)`: Invoked when a new configuration is added to the connector. This is an optional function that can be overridden while setting up a new configuration.
  `on_update_config(selfself, old_config, new_config)`: Invoked when a configuration is updated for the connector. This is an optional function that can be overridden while setting up a configuration edit function.
  `on_delete_config(self, config)`: Invoked when a configuration is deleted from the connector. This is an optional function that can be overridden while setting up a configuration teardown function.
  `on_activate(self, config)` : Invoked when a configuration is activated.
  `on_deactivate(self, config)` : Invoked when a configuration is deactivated.
  `on_teardown(self)` : Invoked when a configuration is deleted. This is an optional function that can be overridden for dismantling a connector.
  You can use these functions to perform specific operations such as starting or stopping of a service at the relevant events. For example, you can use the `on_add_config()` function to start and stop a service when a configuration is added.

playbooks.json

The `playbook.json` file contains any playbook collection that you want to include with your connector. This could be a set of playbooks that demonstrate the usage of your connector.

images

The images directory must have the connector icon files. The 'icon_small_name' and 'icon_large_name' keys in the `info.json` must match the names of these icon files inside the `images` folder. Icon files can be in the `.jpg` or `.png` formats.

Once you have created all the files, bundle them into a *.tgz* file. For example, `tar -czvf sampleConnector.tgz sampleConnector/`.

requirements.txt and packages

If a connector requires additional python libraries, specify the libraries in the `requirements.txt` file.

If there is a dependency on any custom packages or if your instance does not have internet access to download packages from the internet, you can add the packages to the `packages` directory in the connector folder.

During a connector import, the framework first runs `pip install -r requirements.txt` followed by `pip install <package>` for every package in the `packages` directory. The commands are run in a separate thread and import is marked successful even when the dependency installation is still in progress. The dependency install logs are available at `/var/log/cyops/cyops-integrations/pipinstall_<timestamp>.log` on the FortiSOAR instance. If more than five *dependency* install log files get accumulated, the log files that are older than a day get deleted.

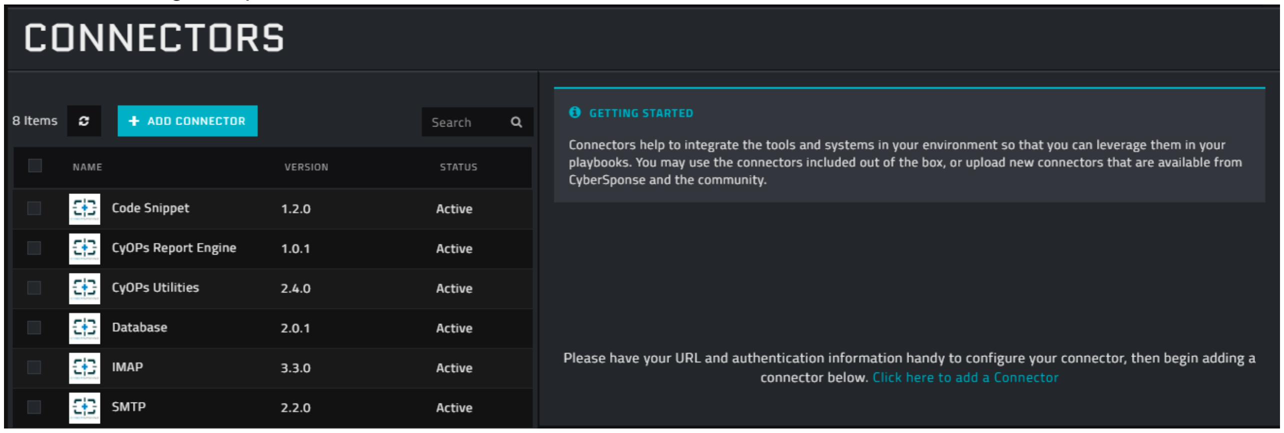
If a dependency install fails, then you can install them again by invoking the REST APIs directly. Contact FortiSOAR Support for more details on the APIs.

Importing a connector into FortiSOAR

Use the "**Connector Store**" to install and configure connectors in FortiSOAR 5.0.0 and later. To install a connector, you must be assigned a role that has a minimum of `Create` access to the `Connectors` module. To configure connectors into FortiSOAR, you must be assigned a role that has a minimum of `Update` access to the `Connectors` module. For more information about the Connector Store and the process of importing connectors into FortiSOAR, see the Introduction to connectors chapter.

Importing a connector into FortiSOAR prior to version 5.0.0

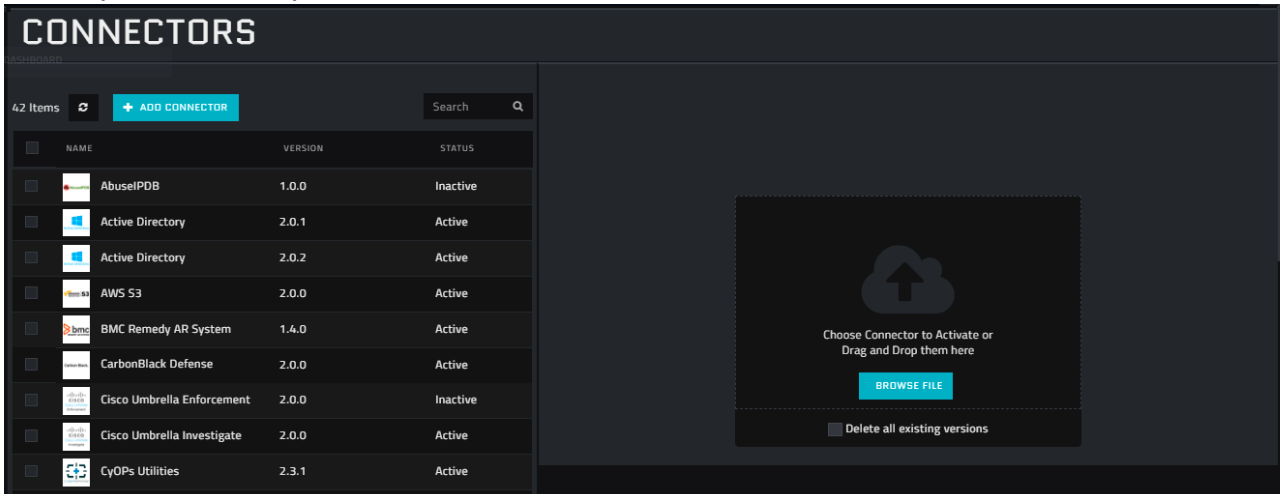
1. Log on to FortiSOAR.
2. On the left navigation pane, click **Automation** > **Connectors** and click **Add Connector**.



NAME	VERSION	STATUS
Code Snippet	1.2.0	Active
CyOPs Report Engine	1.0.1	Active
CyOPs Utilities	2.4.0	Active
Database	2.0.1	Active
IMAP	3.3.0	Active
SMTP	2.2.0	Active

Connectors help to integrate the tools and systems in your environment so that you can leverage them in your playbooks. You may use the connectors included out of the box, or upload new connectors that are available from CyberSponse and the community.

Please have your URL and authentication information handy to configure your connector, then begin adding a connector below. Click here to add a Connector

3. Drag-and-drop the connector package file or click **Browse File** to import the *connector.tgz* file.
   FortiSOAR will prompt you to enter the configuration inputs that you have defined in the `info.json` file.
   Note: You can install different versions of a connector, enabling you to reference a specific version of a connector from a playbook. If you want to replace all previous versions of the connector, ensure that you click the **Delete all existing versions** checkbox while importing the new version of the connector. If you do not click the **Delete all existing versions** checkbox, then a new version of the connector is added. You must ensure that your playbooks reference a correct and existing version of the connector.
   Following is a sample image:
   
   CONNECTORS

   42 Items

NAME	VERSION	STATUS
AbuseIPDB	1.0.0	Inactive
Active Directory	2.0.1	Active
Active Directory	2.0.2	Active
AWS S3	2.0.0	Active
BMC Remedy AR System	1.4.0	Active
CarbonBlack Defense	2.0.0	Active
Cisco Umbrella Enforcement	2.0.0	Inactive
Cisco Umbrella Investigate	2.0.0	Active
CyOPs Utilities	2.3.1	Active

   Choose Connector to Activate or Drag and Drop them here

   Delete all existing versions

   FortiSOAR displays the `Uploading Connector` message and then displays the Connector Configuration popup.
4. To configure the connector, the Connector Configuration popup enter the required configuration details.
   The configuration details and the details of the connector specified in the `info.json` file are stored in the FortiSOAR database.
   Note: You can add multiple configurations for your connector if you have more than one instance of your third-party server in your environment. You must, therefore, add a unique `Name` for each configuration.
   If you have previous versions of a connector and you are configuring a newer version of that connector, with the same configuration parameters, then FortiSOAR fetches the configuration and input parameters of the latest available version of that connector. For example, if you have 1.0.0, 2.0.0, and 2.3.0 versions of the Fortinet FortiSIEM connector and you are configuring the 2.3.0 version of the Fortinet FortiSIEM connector, then while configuring the 2.3.0 version, FortiSOAR will fetch the configuration and input parameters from the 2.0.0 version of the Fortinet FortiSIEM connector. You can review the configuration and input parameters, and then decide to change them or leave them unchanged.
   You can activate or deactivate a configured connector by clicking on the **Activate Connector** or **Deactivate Connector** Link.
   You can check the **Mark As Default Configuration** option to make the selected configuration, the default configuration of this connector, on the particular FortiSOAR instance.
   The `password` type fields in FortiSOAR include encryption and decryption. Passwords are encrypted before saving them into the database and decrypted when they are used in actions. In case of an upgrade, connectors that are already installed will work with stored passwords.
5. To save your configuration, click **Save**.
   To view the list of actions that can be performed by the connector, click the **Actions** tab.
   To view the playbook file that is bundled with the connector, click the **Playbooks** tab.
   You can optionally perform a Health Check by clicking the **Refresh** icon that is present in the `Health Check` bar.
   The Health Check checks if the configuration parameters you have specified are correct and if connectivity can be established to the specified server, endpoint or API.
   If all the details are correct and the connectivity to the server can be established, then on the Connectors page, **Available** is displayed in the health check dialog.
   If any or all the details are incorrect or if the connectivity to the server cannot be established then on the Connectors

page, **Disconnected** is displayed in the health check dialog.
You can also click the **Refresh** icon that is present in the `Health Check` bar to perform the health check at any time.

Reimporting a connector

After importing a connector, any changes done in the python files of the connector automatically get reflected the next time you run a command on the connector. However, if changes are made in the `info.json`, then you need to reimport the connector, using the following command, for the updates to take effect:

```
/opt/cyops-integrations/.env/bin/python /opt/cyops-integrations/integrations/manage.py
reimport_connector -n <connector_name> -cv <connector_version> -migrate
```

If you only want to update the `info.json` changes and not retain the previous connector configuration, then you can omit the `-migrate` attribute as follows:

```
/opt/cyops-integrations/.env/bin/python /opt/cyops-integrations/integrations/manage.py
reimport_connector -n <connector_name> -cv <connector_version>
```

You can also reimport all connectors at a single time using the following command by omitting the `-n <connector_name>` and `-cv <connector_version>` attributes as follows:

```
/opt/cyops-integrations/.env/bin/python /opt/cyops-integrations/integrations/manage.py
reimport_connector -migrate
```

Check_Health function

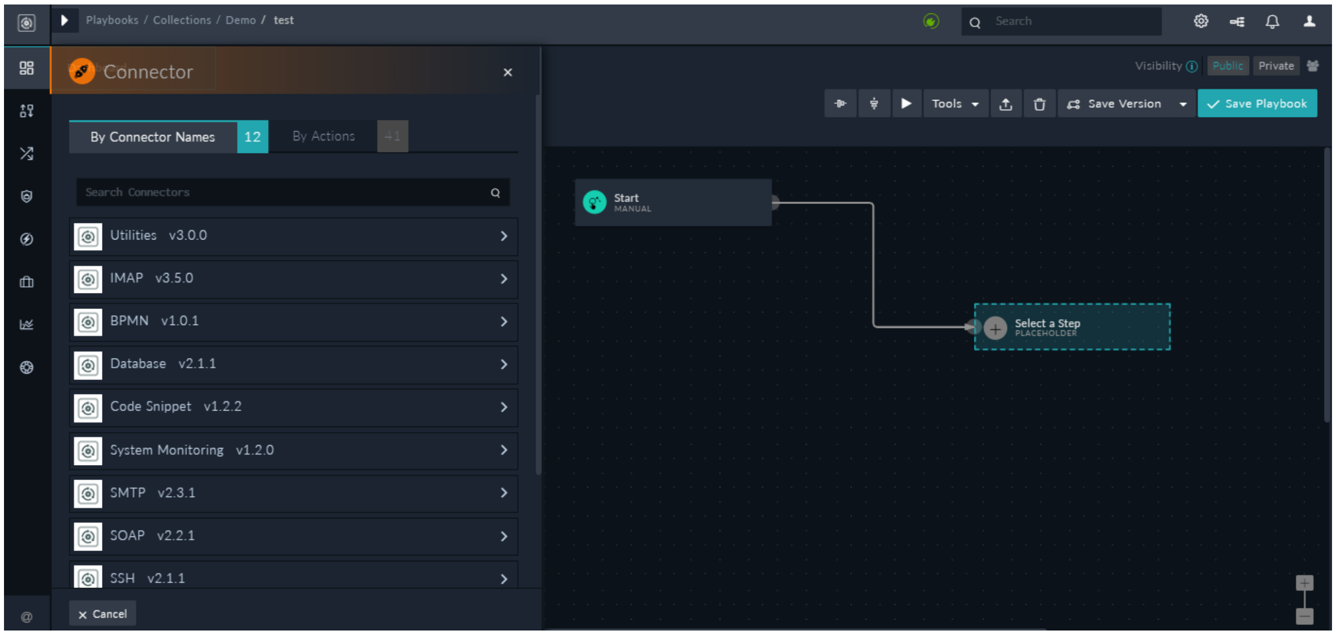
The `check_health` function of the connector is invoked when you click the **Refresh** icon. This function takes the dictionary of the configuration parameters as the input. For example: `{'url': 'https://xyz.com', 'user': 'admin', 'password': 'password'}`.

You must throw the `ConnectorError` from the function if you want the `check_health` function to fail in a given scenario, such as issues with connectivity or with the provided credentials. To throw the `ConnectorError`, you must import it from the `connectors.core.connector` module as follows:

```
from connectors.core.connector import ConnectorError
```

Add connector operation to a playbook

Once you have completed configuring and deploying your connector, you can add a connector operation to a playbook, by adding the connector as a step in the playbook, as shown in the following image:



You can install different versions of a connector, and while adding a connector operation, you specify a specific version of a connector within a Playbook. In case you have installed multiple connectors, and if the version of the connector specified in the playbook is not found, then the playbook by default uses the latest version. FortiSOAR checks for the latest version of the connector in the format "major version.minor version.patch version". For example, version 2.0.2 is a later version than 1.2.0.

The `execute` function of `connector.py` is called when an operation on the connector is invoked. This function takes the dictionary containing the `config`, `operation` and `params` fields. For example:

```
{'config': {'password': password, 'server_url': 'https://xyz.com', 'user': 'admin'},
'params': {'input1': 'value1', 'input2': 'value2'},
'operation': 'function_template'}
```

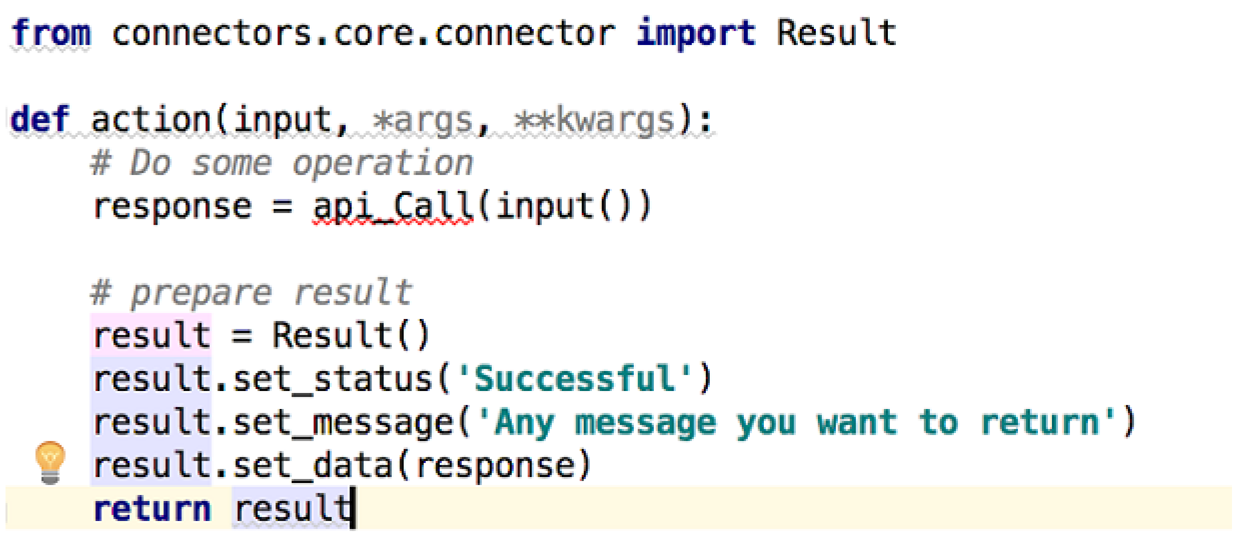
The return from the `execute` function is set in the `results.data` variable of the playbook step. A sample `execute` function is present in the connector.py section.

The `info.json` contains `output_schema`, which defines the keys that are present in the output json on the execution of an operation. The `info.json` contains some common keys. However, the output json can have additional keys based on the input parameters. You can use these json keys to set the input for subsequent Playbook Steps, using Dynamic Values. For more information, see the *Dynamic Values* section in the "Playbooks Guide."

Configuring a connector to return a response

If you want to return a response from any action of a connector, then you can import the `Result` class from the connectors framework to your `connector.py` and then you can set message attributes such as status, message, etc.

Following is the `python` snippet for the same:



```
from connectors.core.connector import Result

def action(input, *args, **kwargs):
    # Do some operation
    response = api_Call(input())

    # prepare result
    result = Result()
    result.set_status('Successful')
    result.set_message('Any message you want to return')
    result.set_data(response)
    return result
```

Updating a connector configuration using the update_connnector_config() function

The `update_connnector_config()` function can be internally called from any connector to update the configuration of any connector by providing the name and version of that connector. Steps to be followed for updating a connector configuration:

1. Import the `update_connnector_config()` function from `/opt/cyops-integrations/integrations/connectors/core/utils.py`.
2. Call the `update_connnector_config` function with the following parameters: `<connector_name>`, `<connector_version>`, `<update_config>`, and `<config_id>`.
   `update_connnector_config(<connector_name>,<connector_version>,<update_config>,<config_id>)`
   For example, `update_connnector_config("imap","3.2.0",{"username":"csdamin","password":"csadminpwd"},"5785130913212321")`
   Notes:
   If you do not provide the `<config_id>`, then the connector configuration that you have marked as default configuration will be updated.
   If you have not marked any connector configuration as the default configuration, and you have also not provided the `<config_id>`, then the following error is raised: `No configuration found to update. Please add a config_id or mark a configuration as the default.` in the `/var/log/cyops/cyops-integrations/connectors.log`. To resolve this error, either mark a connector configuration as the default configuration or provide the `<config_id>`.

Configuring the custom connector to support data ingestion

If your custom connector also supports data ingestion, you require to make the following updates so that your connector works with the ingestion wizard in the FortiSOAR:

1. Update the `info.json` file to contain the following:
   `"ingestion_supported": true`
   Also, specify the ingestion method(s) that is supported by your custom connector. FortiSOAR supports the following ingestion methods:
   `"ingestion_modes": ["scheduled", "notification", "app_push"]`
2. You must include sample ingestion playbooks with your custom connector. You can refer to the sample collection of ingestion-enabled connectors, such as FortiAnalyzer and FortiSIEM, which are included with FortiSOAR. Your sample ingestion playbooks must be created keeping the following points in mind:
   a. Each playbook that contributes to data ingestion must contain the following tags:
      `<connector_name>`, `dataingestion`
   b. **Fetch Playbook**: This playbook fetches sample data and is also used for the actual fetch operation once ingestion is configured. This playbook must be created considering the following:
      Contains an additional tag: `fetch`
      Contains a `Set Variable` step named "Configuration" that defines all user inputs needed for the playbook
      Returns results in "data" keys. `{{vars.result.data}}` of this playbook is what is presented to users as source data on the **Field Mapping** screen in the data ingestion wizard.
   c. **Ingest Playbook**: This is a wrapper playbook that is scheduled or called from the notification service. This playbook generally calls the fetch playbook as a reference and then loops over the result creating "alert" records in FortiSOAR.
      The step that creates records should be named "Create Record". The ingestion wizard reads this step and displays the default mappings from the source data to the record created in FortiSOAR. If the step uses a 'bulk' Create Record step, the mappings are shown using the `item` variable. This playbook must have the following additional tags: `fetch`, `create`.
   d. **Create Record Playbook**: If record creation is a multi-step process and cannot use the single "Create Record" step in bulk, then you can move record creation to a separate playbook that is invoked in a loop after fetch. If this is the case, then the "Create Record" playbook must have the `create` tag and this tag must not be present in the "Ingest" playbook. This playbook must be created considering the following:
      Defines an input parameter "sourcedata"
      Contain only one Create Record and the step name must also be named "Create Record"
      "Create Record" step works on the "sourcedata" input
      Note: Multiple tags can be applied on the same playbook. However, you need to ensure that the "fetch" playbook does not also create records, since otherwise it will also be called for fetching sample data.

FortiSOAR Connector SDK

The FortiSOAR Connector SDK consists of a simulator for writing and testing your connector. You can use this CLI to help you develop your custom connector, without having a ForiSOAR™ instance.

For information on how to setup and use the FortiSOAR Connector SDK, see Fortinet Developer Network (FNDN). You can also download the Connector SDK from FNDN and use it to develop a custom connector. You must log into FNDN to access the Connector SDK.

Data Ingestion

FortiSOAR has a dedicated data ingestion wizard that facilitates data ingestion from external SIEM solutions and other third-party sources like threat intelligence platforms, email solutions, etc. The wizard also takes care of scheduling of data ingestion into FortiSOAR, if the connector is enabled for scheduling.

The Data Ingestion wizard eases data ingestion configuration and mapping of fields between the two systems. The wizard assists in fetching sample data from the source, mapping fields from the source data into a FortiSOAR module and sets up an ingestion schedule if required and if the connector is enabled for scheduling. Based on the inputs provided in the wizard, the system dynamically generates the ingestion workflows without the user being exposed to the details of playbooks and easing the process of configuring ingestion.

The Data Ingestion wizard allows mapping of dropdown (picklist) fields and items such that you can map picklist items to a range of values, lesser than, and greater than values. The Data Ingestion wizard allows you to define the frequency at which you want to pull content into FortiSOAR from third-party integration, such as SIEMs that allow you to define the polling frequency. By default, the frequency of pulling content is set at five minutes.

FortiSOAR and its connectors come with sample playbooks for ingesting data from various SIEM sources, Threat Intelligence Platforms, Vulnerability Management Tools, Configuration Management Databases, etc. Connectors that you can use to ingest data through this wizard are enabled by default in FortiSOAR.

FortiSOAR 6.0.0 has enhanced the data ingestion as follows:

- Support for fetching data for each configuration of your connector, i.e., if you have two configurations present in your connector, then you can pull ingestion data for each configuration.
- Support for multiple queries for pulling data based on your requirement; however, you must specify one query to schedule pulling of data from the connector into FortiSOAR.
- Enhanced the UI of the data ingestion wizard to make it easier to work with the wizard.
- Added a **Data Ingestion** tab on the `Connectors` page that monitors your data ingestion and provides information on which connectors are configured for using the Data Ingestion Wizard, as well as other information such as what is the status of a configuration, what is the schedule for ingestion, when data was last pulled using that configuration, etc.
- Added support for inserting or upserting records in bulk during data ingestion, which improves the performance since all the records are created or upserted in one request.

Modes of Data Ingestion

Data Ingestion can work in any of the following three modes:

- **Notification Based**: Some connectors such as, IMAP, Exchange, Syslog, etc have a Notification Service that gets instantly notified when a message arrives on the server. The notification service, in turn, triggers a FortiSOAR playbook to create FortiSOAR records from the fetched data.
- **Schedule Based**: Some connectors such as QRadar, Anomali, ServiceNow, etc use the Fetch APIs of the integration, i.e., for example, the fetch API of Anomali, along with a user-defined query to fetch data from the product into FortiSOAR. These fetch playbooks are scheduled to run by default, every 5 mins or can also be scheduled to run according to a user-defined interval.
  Note that most integrations that support a Notification Based ingestion also support a Schedule Based ingestion.

You must, however, configure only one of the two since otherwise both would pull the data from the same source and there might be data loss due to conflicts.

- **App Push**: Some connectors such as Splunk have an add-on that is provided by FortiSOAR and which can be installed on the server side to push data from the integration to FortiSOAR. The add-on is configured with a user-password or appliance-based authentication in FortiSOAR and it triggers FortiSOAR playbooks to create the records in FortiSOAR.

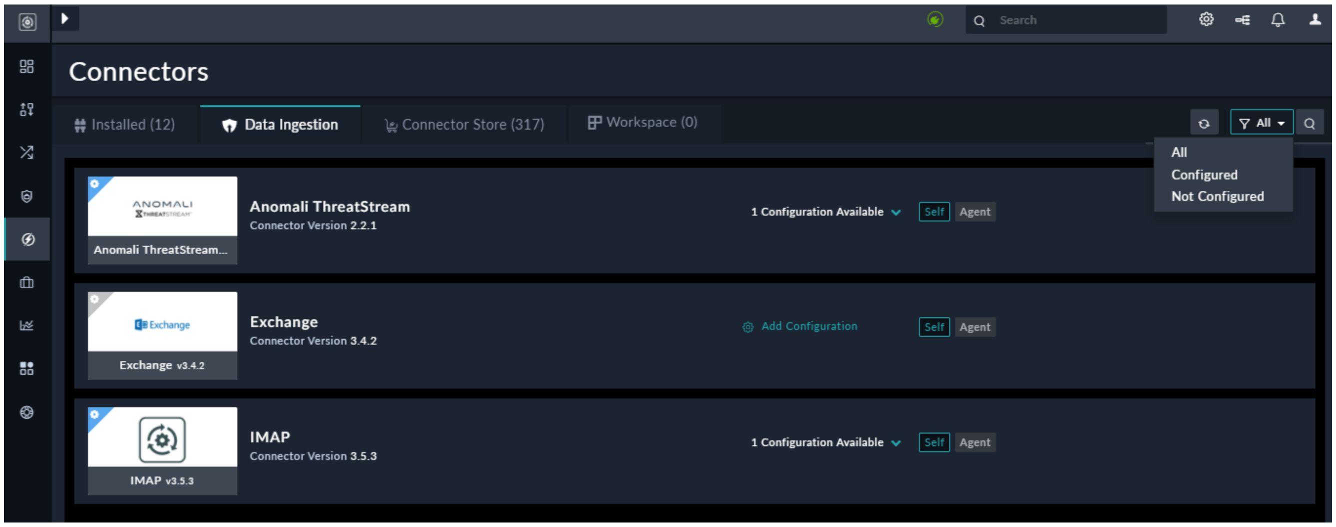
Permissions required for using the Data Ingestion Wizard

To use data ingestion, you must be assigned a role that has `Read` and `Update` permissions on the `Connectors` module and `Create`, `Read`, `Update`, and `Execute` permissions on the `Playbooks` module. If you want to schedule the data ingestion, then you also require to have `Create`, `Update` and `Read` permissions on the `Schedules` module.

Data Ingestion Wizard

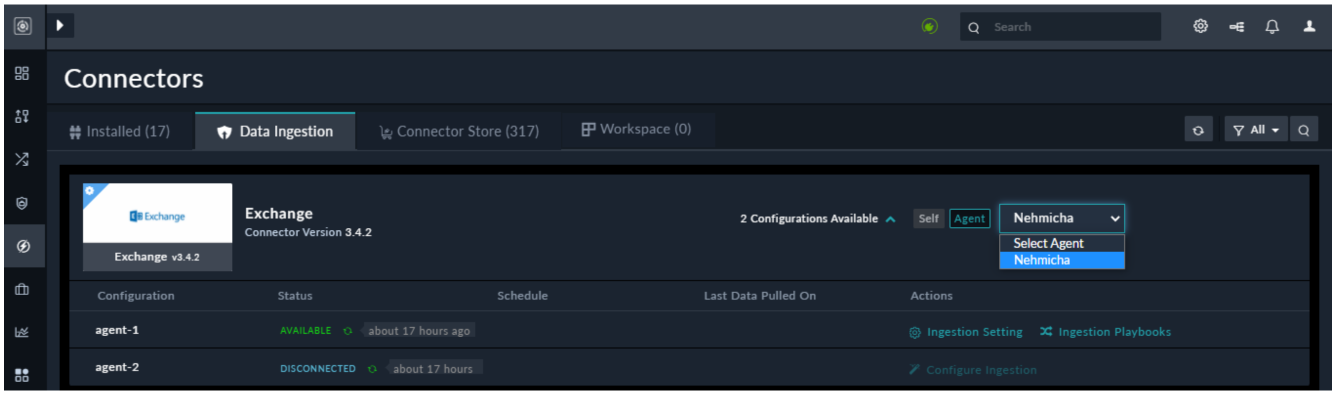
The **Data Ingestion** tab displays connectors that are configured for using the Data Ingestion Wizard, i.e., you can use the data ingestion wizard to ingest data into FortiSOAR for these connectors.

To view the data ingestion tab, log on to FortiSOAR and on the left navigation pane, click **Automation** > **Connectors**. Click the **Data Ingestion** tab, which will display all the connectors that are enabled for data ingestion, along with the number of configurations available for that connector.



You can filter the display on the `Data Ingestion` page to show all the connectors that are enabled for data ingestion, whether configured or not, by clicking the **All** (default) filter. To view connectors that are enabled for data ingestion, but not configured, click the **Not Configured** filter, and to view connectors that are enabled for data ingestion that are configured, click the **Configured** filter. You can search for a connector by connector name in the **Search by connector name** field. In case of connectors that are not configured, you can click the **Add Configuration** link to open the Connector Configuration page of that connector enabling you to configure the connector.

From version 6.4.3 onwards, support is added for running the data ingestion wizard on agent configurations. To display the details of the configurations, click the `<number of configurations> configurations available` link. To view the configuration details of the current node, click **Self**. To view the agent configuration details, click **Agent** and then from the drop-down list, select the name of the agent whose configuration details you want to view, as shown in the following image:

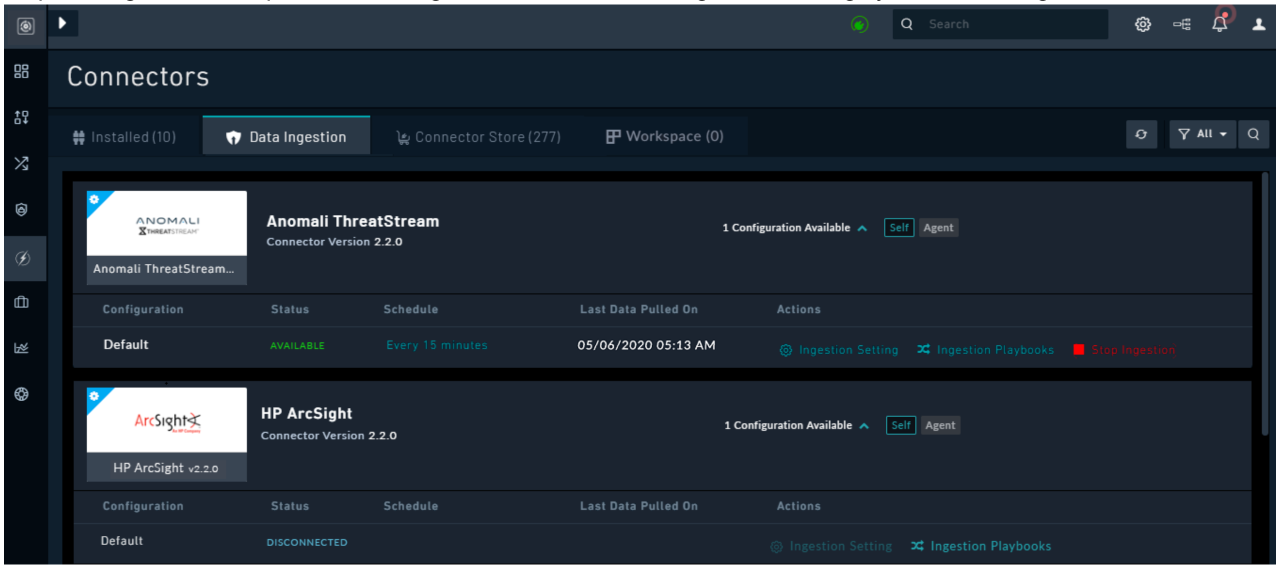


Exchange
Connector Version 3.4.2
2 Configurations Available
Agent: Nehmicha

Configuration	Status	Schedule	Last Data Pulled On	Actions
agent-1	AVAILABLE about 17 hours ago			Ingestion Setting, Ingestion Playbooks
agent-2	DISCONNECTED about 17 hours			Configure Ingestion

Details that are displayed for the configurations are as following:

- **Status** of a configuration, whether it is **Available** or **Disconnected**.
- **Schedule** for running the ingestion, i.e., what is the frequency at which the third-party integration will be polled to ingest data into FortiSOAR.
- **Last Data Pulled on** that contains the datetime when the data was last pulled into FortiSOAR using this configuration.
- **Actions** contains the following options:
  Configure Ingestion: Clicking this link displays the Data Ingestion Wizard. This option is displayed when you are configuring data ingestion for the first time for a particular ingestion.
  Ingestion Settings: Clicking this link displays the Data Ingestion Wizard with the configured ingestion settings. This option is displayed when you have already configured data ingestion for a particular configuration.
  Ingestion Playbooks: Clicking this link opens the ingestion playbooks collection in a new window.
  Start Ingestion / **Stop Ingestion**: Clicking this link either starts an ingestion for a newly-added configuration or stops an ingestion for a particular configuration based on the ingestion settings you have configured.



Anomali ThreatStream
Connector Version 2.2.0
1 Configuration Available

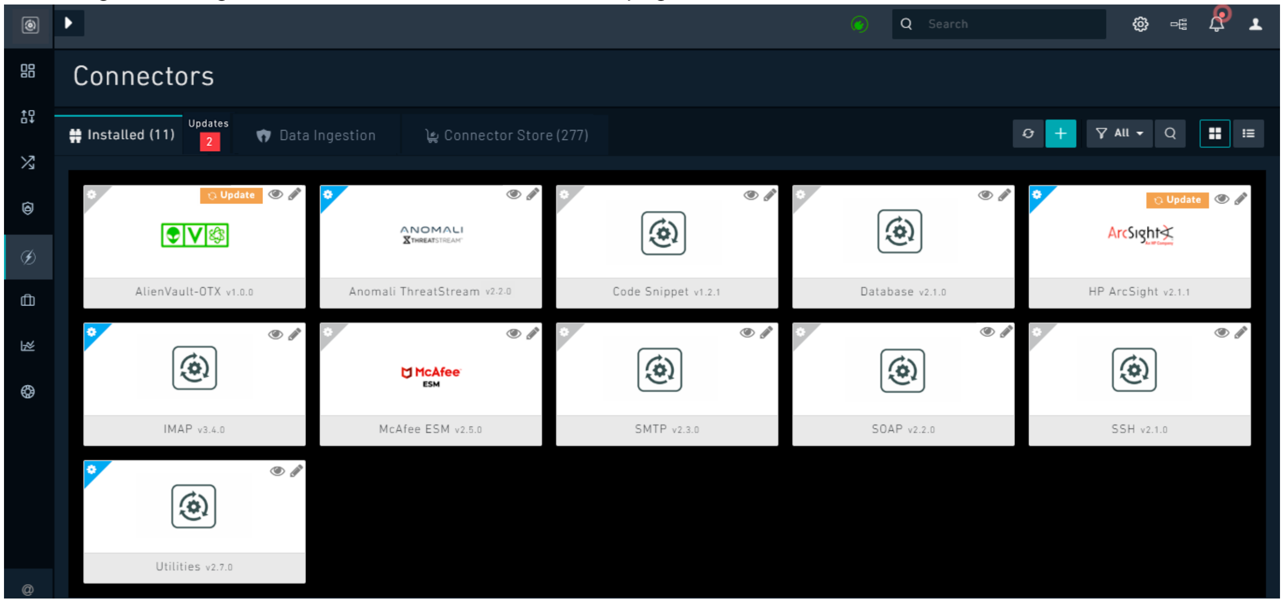
Configuration	Status	Schedule	Last Data Pulled On	Actions
Default	AVAILABLE	Every 15 minutes	05/06/2020 05:13 AM	Ingestion Setting, Ingestion Playbooks, Stop Ingestion

HP ArcSight
Connector Version 2.2.0
1 Configuration Available

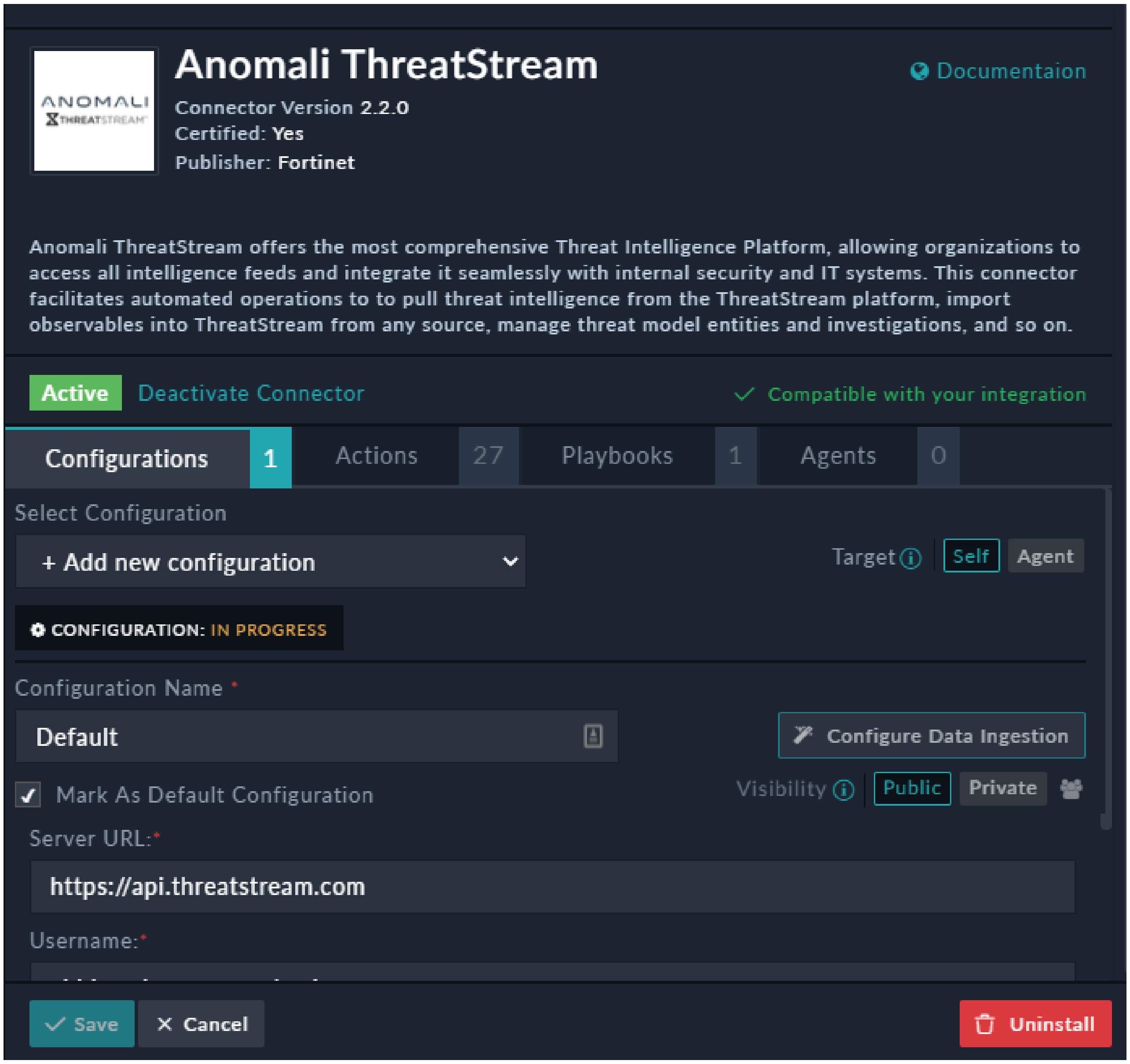
Configuration	Status	Schedule	Last Data Pulled On	Actions
Default	DISCONNECTED			Ingestion Setting, Ingestion Playbooks

If you have upgraded to FortiSOAR 6.0.0 or later from an earlier version, then before you reconfigure ingestion for the same connector configuration, you must deactivate the earlier ingestion playbooks that are present in the ingestion collection for the connector. The links to the ingestion playbooks that were created prior to the upgrade will be present on the **System Fixtures** page in the "Ingestion Playbooks" section will not be visible in the Data Ingestion tab (new in version 6.0.0) of the "Connectors" page. If your data ingestion is schedule-based, then you must also stop or delete the earlier schedules for the connector.

Process of ingesting data using the Data Ingestion Wizard

1. Log on to FortiSOAR.
2. On the left navigation pane, click **Automation** > **Connectors** > **Installed**.
   On `Installed Connectors` page, you will see the list of installed connectors, either in grid/list view or in the card view. You will also see an **Data Ingestion** tab on the `Connectors` page, for more information, see Data Ingestion.
   Following is an image of the `Installed Connectors` page in the Card view:
   

3. Click a connector using which you want to ingest data. For our example, we have chosen Anomali ThreatStream.
4. On the `Connector Configuration` pane, from the **Select Configuration** drop-down list select the configuration for which you want to configure the ingestion, and then click **Configure Data Ingestion**.



Anomali ThreatStream

Documentaion

Connector Version 2.2.0
Certified: Yes
Publisher: Fortinet

Anomali ThreatStream offers the most comprehensive Threat Intelligence Platform, allowing organizations to access all intelligence feeds and integrate it seamlessly with internal security and IT systems. This connector facilitates automated operations to to pull threat intelligence from the ThreatStream platform, import observables into ThreatStream from any source, manage threat model entities and investigations, and so on.

Select Configuration: + Add new configuration

Target: [✓ Self] Agent

CONFIGURATION: IN PROGRESS

Configuration Name *: Default

✓ Mark As Default Configuration

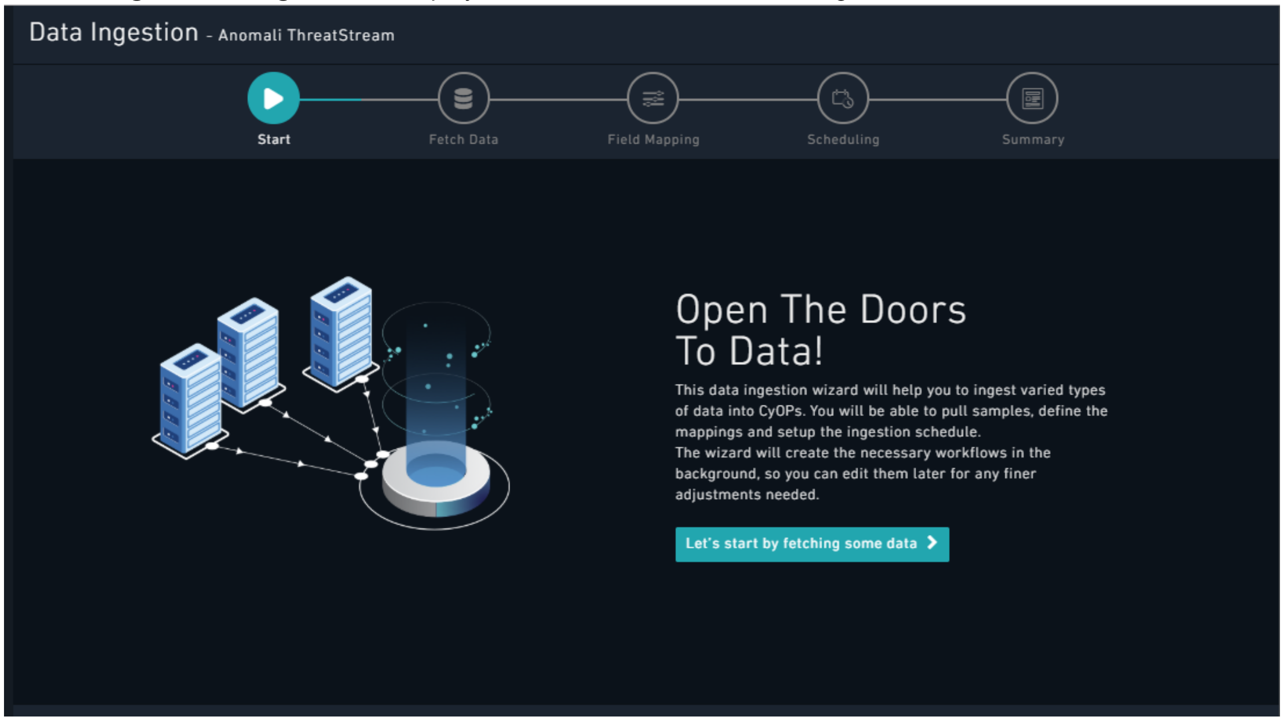
Visibility: [✓ Public] Private

Server URL:*: https://api.threatstream.com

Username:*

Only connectors that have been enabled for data ingestion will have the **Configure Data Ingestion** button. To ingest data, your connector must be in the **Available** state.

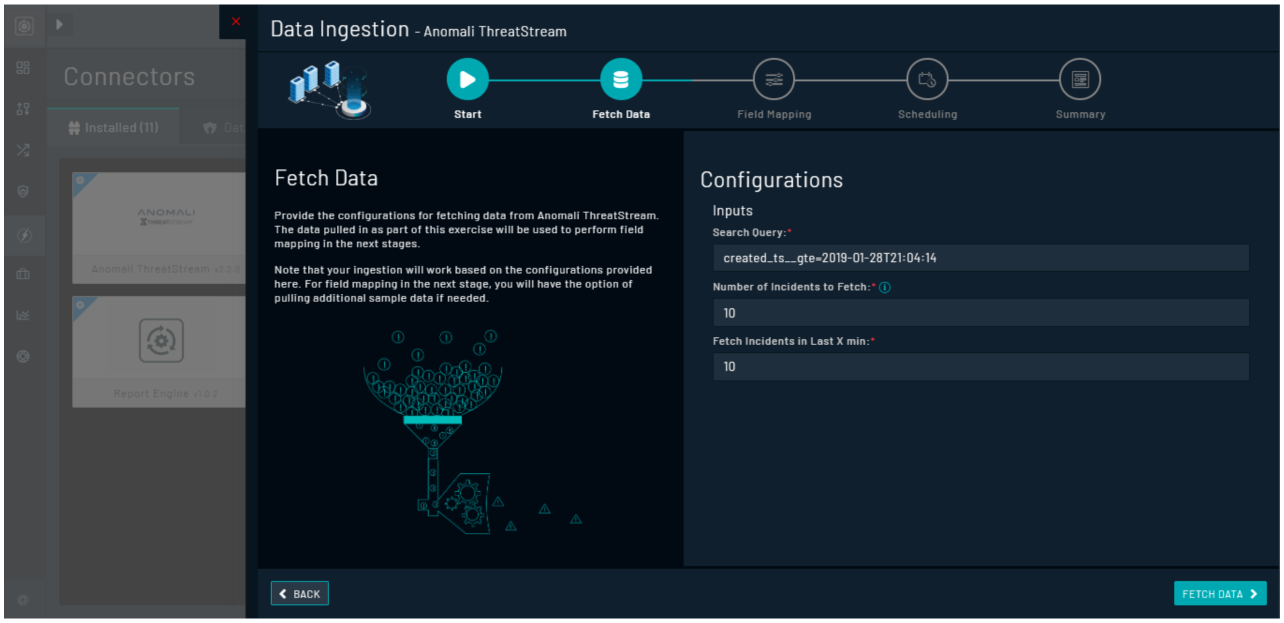
5. Click **Configure Data Ingestion** to display the `Start` screen of the Data Ingestion Wizard.


Data Ingestion - Anomali ThreatStream
Start
Fetch Data
Field Mapping
Scheduling
Summary
Let's start by fetching some data

Open The Doors To Data!

This data ingestion wizard will help you to ingest varied types of data into CyOPs. You will be able to pull samples, define the mappings and setup the ingestion schedule.
The wizard will create the necessary workflows in the background, so you can edit them later for any finer adjustments needed.

6. Click the **Let's Start..** button to display the `Fetch Sample Data` screen.
7. Sample data is required to create a field mapping between your connector data and FortiSOAR. The sample data is pulled from connector actions or ingestion playbooks. By default, FortiSOAR ingests data using an "Ingestion Playbook" that is included by default with each connector that is enabled for data ingestion.
   A connector would generally require some configuration details, which you will require to fill in on this screen. For example, in the case of Anomali ThreatStream, in the **Search Query** field, enter the query based on which you want to fetch data into FortiSOAR. In the **Number of Incidents to Fetch** field, enter the number of incidents that will be fetched from Anomali ThreatStream, for example **10**. In the **Fetch Incidents in Last X Min**, enter the number of minutes for which indicators that will be fetched from Anomali ThreatStream, for example **10**, in this case the data ingestion wizard will fetch indicators created in the last 10 minutes in Anomali ThreatStream, and then click **Fetch Data**.



Data Ingestion - Anomali ThreatStream

Fetch Data

Provide the configurations for fetching data from Anomali ThreatStream. The data pulled in as part of this exercise will be used to perform field mapping in the next stages.

Note that your ingestion will work based on the configurations provided here. For field mapping in the next stages, you will have the option of pulling additional sample data if needed.

Configurations

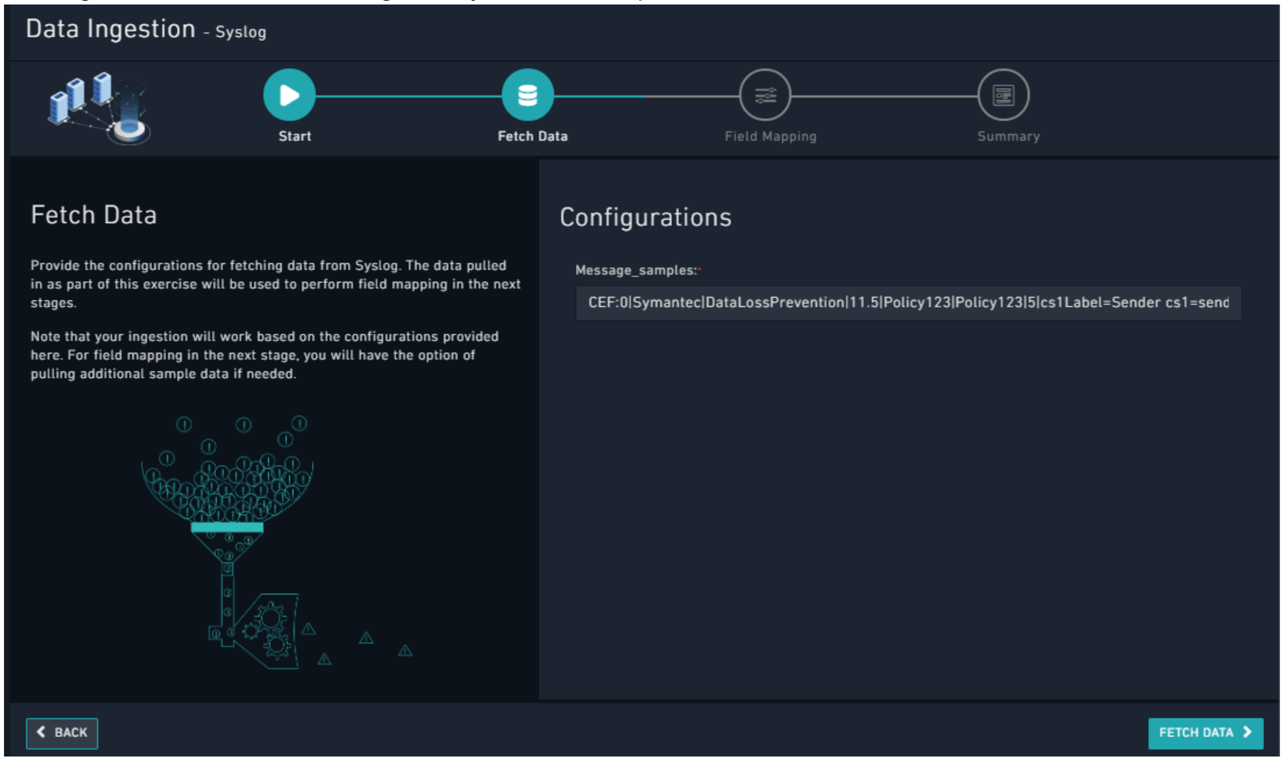
Inputs

Search Query:* created_ts__gte=2019-01-28T21:04:14

Number of Incidents to Fetch:* 10

Fetch Incidents in Last X min:* 10

Another example is **Syslog** where you will see the **Message_samples** field in which you can add samples of messages in the CEF Format using which you want to map the fields:



Data Ingestion - Syslog

Fetch Data

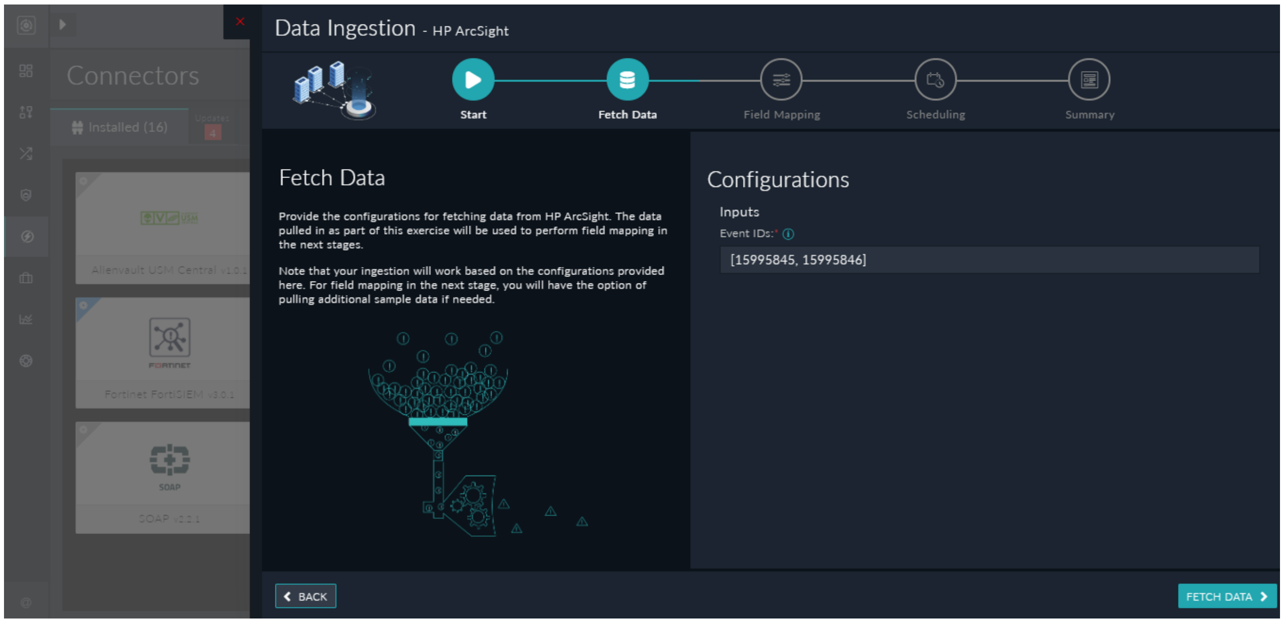
Provide the configurations for fetching data from Syslog. The data pulled in as part of this exercise will be used to perform field mapping in the next stages.

Note that your ingestion will work based on the configurations provided here. For field mapping in the next stage, you will have the option of pulling additional sample data if needed.

Configurations

Message_samples:* CEF:0|Symantec|DataLossPrevention|11.5|Policy123|Policy123|5|cs1Label=Sender cs1=send

Or, as is the case of **HP ArcSight**, you will see the **Event IDs** field, in which you require to specify a comma-separated list of valid Event IDs using which you want to map the fields.



Data Ingestion - HP ArcSight

Fetch Data

Provide the configurations for fetching data from HP ArcSight. The data pulled in as part of this exercise will be used to perform field mapping in the next stages.

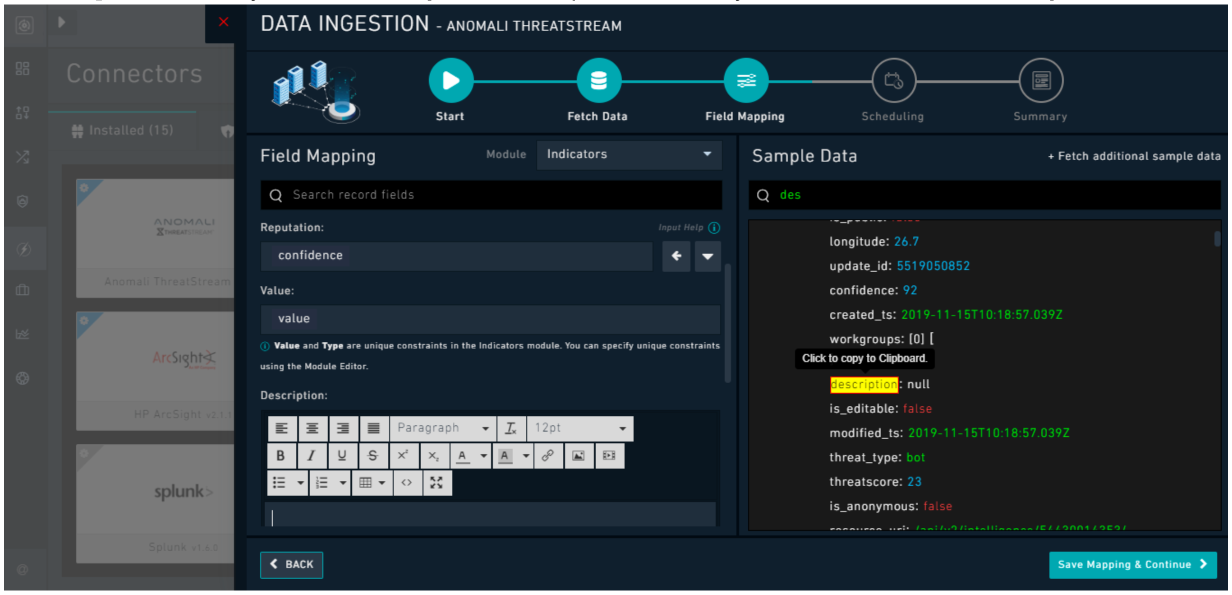
Note that your ingestion will work based on the configurations provided here. For field mapping in the next stage, you will have the option of pulling additional sample data if needed.

Configurations

Inputs

Event IDs:* [15995845, 15995846]

8. On the **Field Mapping** screen, map the fields of the sample data to the fields present in FortiSOAR as follows:
   a. The Field Mapping screen displays the Sample Data on the right side of the screen and the Field Mapping (FortiSOAR fields) on the left side of the screen. The sample data is in the form of a Key-Value pair.
   From the **Module** drop-down list that appears next to Field Mapping, select the FortiSOAR module for which you want to map the fields. The default module will be already be selected, for example, **Indicators**.
   Note: If you select any module other than the default module, you will require to remap all the fields.
   b. From the **Sample Data** fields, map the fields onto the fields present in the **Indicators** module as follows:
   Important: Some fields such as Name and some picklists can come pre-mapped with their keys. You do not require to re-map these fields unless you want to override their default values. You can search for fields in the record and in the sample data.
   To map a field, click the key in the sample data to add jinja for the field. For example, `description` from the Anomali ThreatStream sample data can be mapped to `Description` field in FortiSOAR by clicking `description`. Once you click **description** in Sample Data, its key will be added to the **Description** field:



DATA INGESTION - ANOMALI THREATSTREAM

Field Mapping Module Indicators

Reputation: confidence

Value: value

Value and Type are unique constraints in the Indicators module. You can specify unique constraints using the Module Editor.

Description:

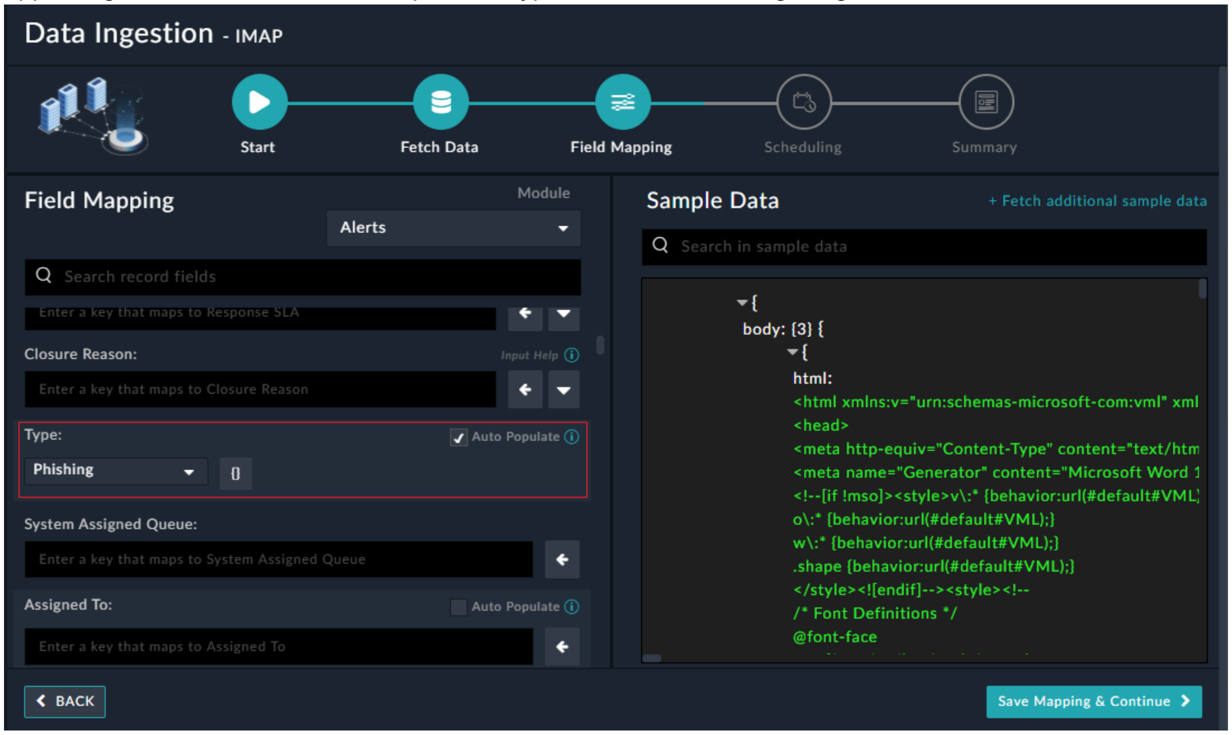
Sample Data

des

```
longitude: 26.7
update_id: 5519050852
confidence: 92
created_ts: 2019-11-15T10:18:57.039Z
workgroups: [0] [
description: null
is_editable: false
modified_ts: 2019-11-15T10:18:57.039Z
threat_type: bot
threatscore: 23
is_anonymous: false
```

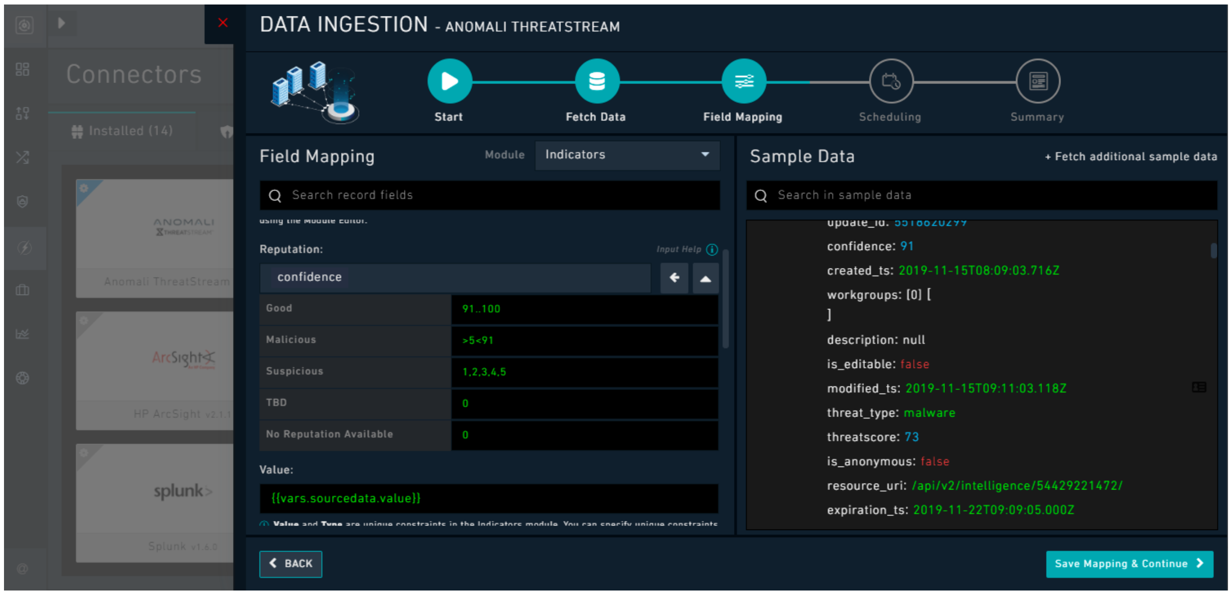
To view the key of a field, double click in the field, for example if you click the newly-mapped description field, its key `{{vars.sourcedata["description"]}}` will be displayed. Once the key (which is in jinja) is

displayed, this field will always be displayed with the key value.
Note: If bulk insert or upsert is supported for the connector, for example, IMAP, then the key value changes from `{{vars.sourcedata["name of the field"]}}` to `{{vars.item["name of the field"]}}`. For example, `{{vars.item["description"]}}`.
From version 7.0.0 onwards, if your administrator has enabled any 'Lookup' or 'Picklist' type of field to accept the values generated from the recommendation engine, then you will see an **Auto populate** checkbox appearing beside this field. For example, the 'Type' field in the following image:



```
{
  body: {3} {
    {
      html:
      <html xmlns:v="urn:schemas-microsoft-com:vml" xml
      <head>
      <meta http-equiv="Content-Type" content="text/htm
      <meta name="Generator" content="Microsoft Word 1
      <!--[if !mso]><style>v\:* {behavior:url(#default#VML)
      o\:* {behavior:url(#default#VML);}
      w\:* {behavior:url(#default#VML);}
      .shape {behavior:url(#default#VML);}
      </style><![endif]--><style><!--
      /* Font Definitions */
      @font-face
```

If you select the **Auto populate** checkbox, and users have not specified any values for such fields, then the value of such fields get auto-populated with the values from the recommendation engine that is based on learning from past similar records.
Picklists can come pre-populated with their defined defaults, if they are available. You can also map picklists using the wizard.
To map a picklist, such as **Reputation**, select the picklist whose value you want to map. For example, `confidence` from the Anomali ThreatStream sample data can be mapped to the `Reputation` picklist in FortiSOAR by clicking `confidence`. FortiSOAR would already have mapped this picklist by default, and it will appear as `confidence` whose key value is `{{vars.sourcedata["confidence"]}}`. Next, you must map the items of the Reputation picklists. To view and map the items of the Reputation picklist in FortiSOAR, click the down arrow (**v**). The Reputation picklist has items such as Good, Malicious, Suspicious, TBD, and No Reputation Available. These picklist items can be mapped as per the values defined in the source. You can map more than one picklist item in FortiSOAR can map to a single value in the source. For example, you can map both **TBD** and **No Reputation Available** to `0` in FortiSOAR. You can also map one picklist item in FortiSOAR to two values or more values in the source, for example, Suspicious can be 1,2,3,4,5. In this case, if the **confidence** field in Anomali ThreatStream has value 1, 2, 3, 4, or 5, then all cases with confidence 1 to 5 will be mapped to **Suspicious** reputation in FortiSOAR. You can also specify mapping of a range of values, lesser than, and greater than values. For example, **Good** can be mapped `91..100` in FortiSOAR, which means that if the **confidence** field in Anomali ThreatStream contains any value between 91 to 100 they will be mapped to **Good** in FortiSOAR. Similarly, if you map **Malicious** to `>5<91` in FortiSOAR this means that if the **confidence** field in Anomali ThreatStream contains any value greater than 5 and less than 91, then they will be mapped to **Malicious** reputation in FortiSOAR.



Reputation:

confidence

Good	91..100
Malicious	>5<91
Suspicious	1,2,3,4,5
TBD	0
No Reputation Available	0

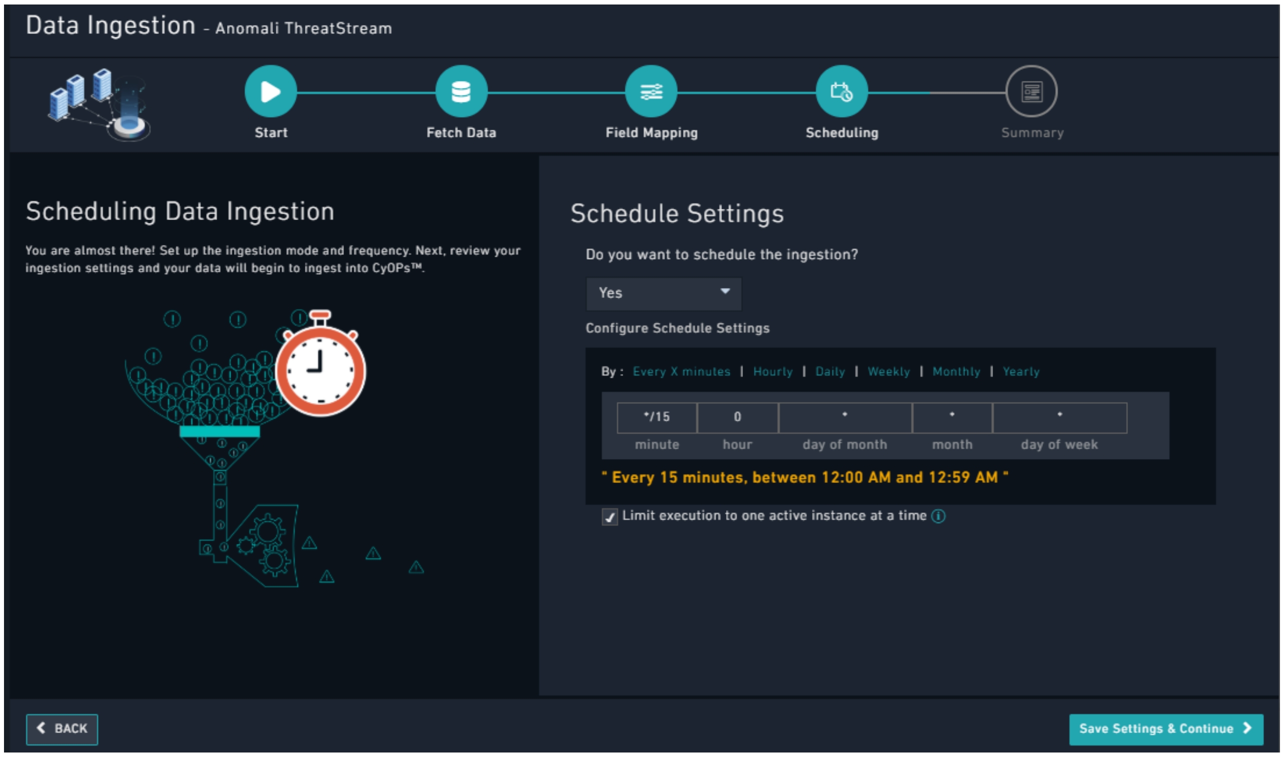
Value:

{{vars.sourcedata.value}}

```
confidence: 91
created_ts: 2019-11-15T08:09:03.716Z
workgroups: [0] [
]
description: null
is_editable: false
modified_ts: 2019-11-15T09:11:03.118Z
threat_type: malware
threatscore: 73
is_anonymous: false
resource_uri: /api/v2/intelligence/54429221472/
expiration_ts: 2019-11-22T09:09:05.000Z
```

Note: Multiple expressions of type "number" must be separated by a space. Also, in case of a range there must be two dots between the numbers.
You can also specify string values in a picklist such as Open, In Progress, Closed, etc. Multiple expressions of type "string" must be comma-separated.
In case you want to use another query to fetch additional data in order to create comprehensive mapping, click the **+ Fetch additional sample data** link that appears in the Sample Data header. Clicking the **+ Fetch additional sample data** link opens the Configurations dialog in which you can change the configuration such as, updating the **Search_query** and click **Fetch Data**. The data ingestion wizard fetches the data based on the updated configuration in a new page in the `Sample Data` section. You can continue mapping based on the newly fetched data.
Once you are satisfied with the mappings, click **Save Mapping & Continue**.

9. (Optional) If your connector is enabled for scheduling, such as Anomali ThreatStream, then you will be shown the **Scheduling** screen, using which you can specify the schedule for data ingestion from the connector into FortiSOAR, i.e., you can specify the polling frequency to a third-party integration, such as SIEM, so that the content gets pulled from the third-party integration into FortiSOAR. By default, scheduling is set to pull data every 5 minutes.
To configure scheduling, from the **Do you want to schedule the ingestion?** drop-down list, select **Yes**.
In the `Configure Schedule Settings` section, specify the Cron expression for the schedule. For example, if you want to pull data from Anomali ThreatStream every 15 minutes, click **Every X Minute** and in the minute box enter `*/15`.



Data Ingestion - Anomali ThreatStream

Start | Fetch Data | Field Mapping | Scheduling | Summary

Scheduling Data Ingestion

You are almost there! Set up the ingestion mode and frequency. Next, review your ingestion settings and your data will begin to ingest into CyOPs™.

Schedule Settings

Do you want to schedule the ingestion? Yes

Configure Schedule Settings

By : Every X minutes | Hourly | Daily | Weekly | Monthly | Yearly

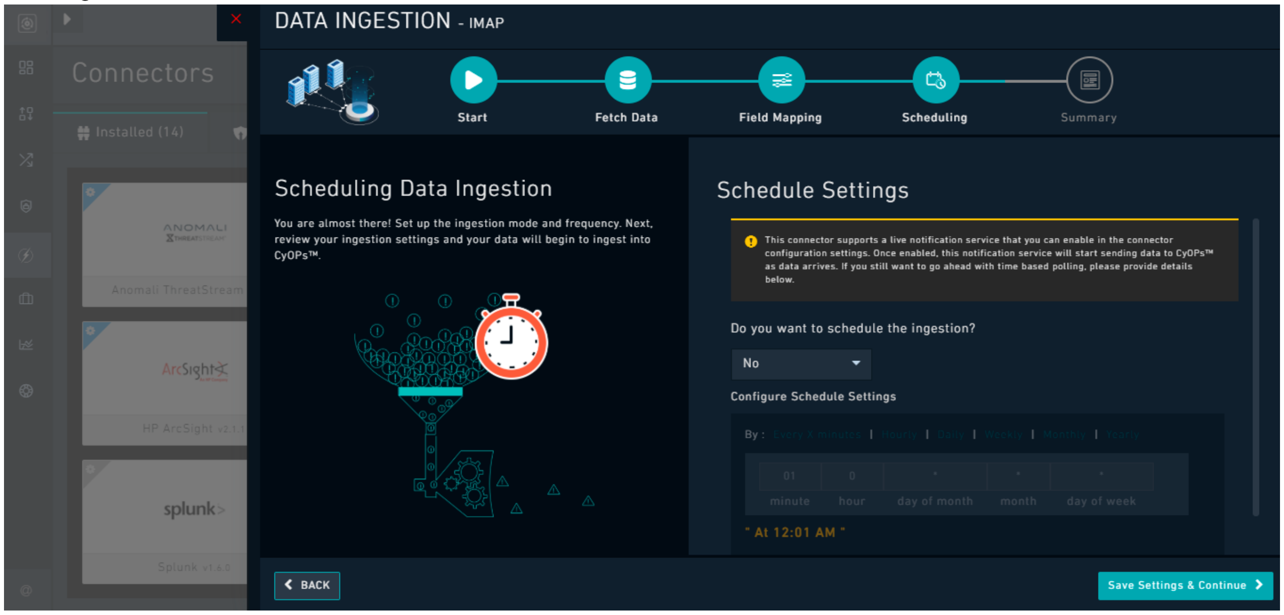
minute	hour	day of month	month	day of week
*/15	0	*	*	*

" Every 15 minutes, between 12:00 AM and 12:59 AM "

✓ Limit execution to one active instance at a time

BACK | Save Settings & Continue

Note: If you select the **Limit execution to one active instance at a time** checkbox, then do not rerun the workflow (schedule) if the previous schedule is still running. Some connectors such as IMAP support data ingestion using both a scheduled-based pull as well as instant notification using the Notification Service. You must, however, configure only one of the two, since both would otherwise pull the data from the same source and there could be data loss due to conflicts. Therefore, the `Scheduling` screen for the IMAP connector displays the following warning:



DATA INGESTION - IMAP

Start | Fetch Data | Field Mapping | Scheduling | Summary

Scheduling Data Ingestion

You are almost there! Set up the ingestion mode and frequency. Next, review your ingestion settings and your data will begin to ingest into CyOPs™.

Schedule Settings

This connector supports a live notification service that you can enable in the connector configuration settings. Once enabled, this notification service will start sending data to CyOPs™ as data arrives. If you still want to go ahead with time based polling, please provide details below.

Do you want to schedule the ingestion? No

Configure Schedule Settings

By : Every X minutes | Hourly | Daily | Weekly | Monthly | Yearly

minute	hour	day of month	month	day of week
01	0	*	*	*

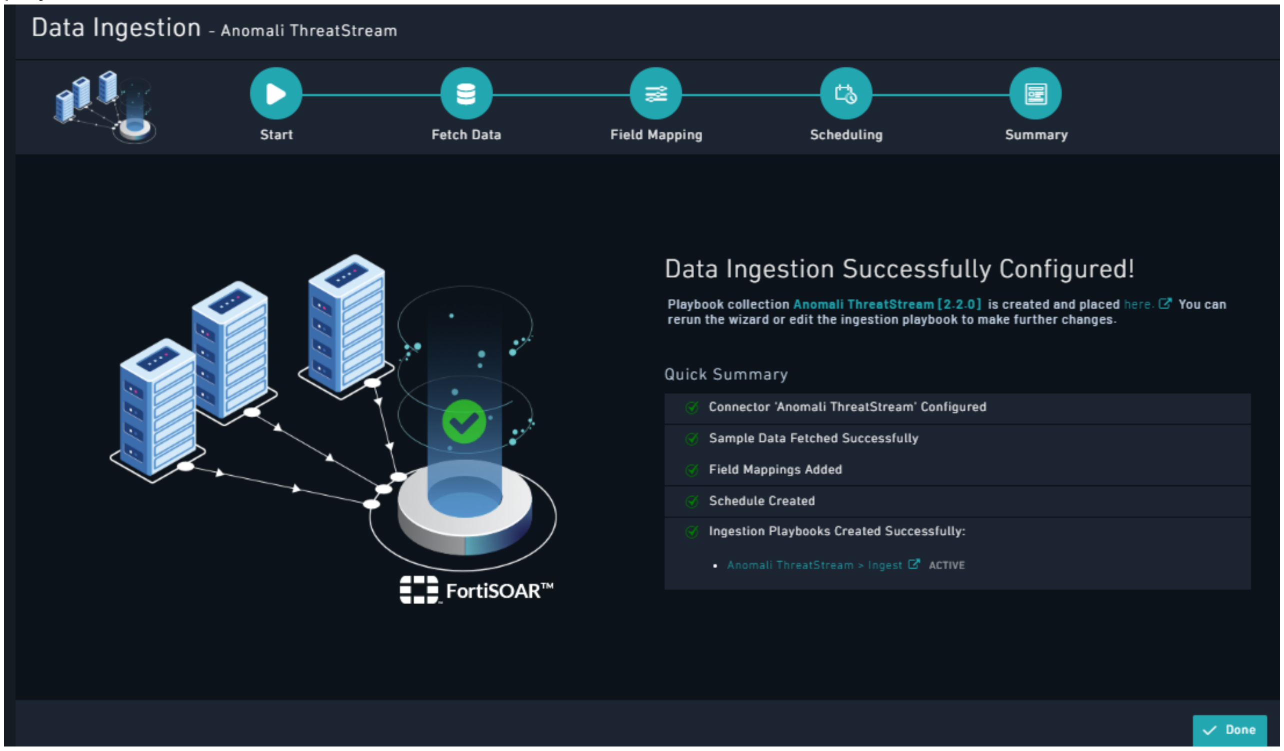
" At 12:01 AM "

BACK | Save Settings & Continue

Only if you want to ingest data using a scheduled-based pull, then you should use the `Scheduling` screen to specify the schedule for data ingestion from the IMAP connector into FortiSOAR.

Once you are satisfied with the scheduling, click **Save Settings & Continue**.

10. The **Summary** screen displays a summary of the mapping done, and it also contains links to the Ingestion playbooks.



Data Ingestion Successfully Configured!

Playbook collection Anomali ThreatStream [2.2.0] is created and placed here. You can rerun the wizard or edit the ingestion playbook to make further changes.

Quick Summary

- Connector 'Anomali ThreatStream' Configured
- Sample Data Fetched Successfully
- Field Mappings Added
- Schedule Created
- Ingestion Playbooks Created Successfully:
  - Anomali ThreatStream > Ingest ACTIVE

Click **Done** to complete the data ingestion and exit the Data Ingestion Wizard.

Notes for Data Ingestion Playbooks:

- The **Summary** step contains links to playbooks that contain the data mapping based on the mappings you have specified in the wizard. You can click the links and open the playbook in the Playbook Designer.
  For example, to open the playbook that fetches data from the Anomali ThreatStream connector, click **Anomali ThreatStream > Ingest** link, which opens the `Anomali ThreatStream > Ingest` playbook in the Playbook Designer.
- You can open the Data Ingestion Playbooks collection by clicking **here** in the Summary step.
- You can modify these data ingestion playbooks or the mappings in the playbooks using the Playbook Designer and those modifications are reflected back in the mappings, i.e., if you open the "Data Ingestion Wizard" again you will see the modified content.

Notes for Data Ingestion

- If you are using the IMAP connector for data ingestion, then you must ensure that you configure either a scheduled-based pull, using Scheduling in the Data Ingestion Wizard, or the instant notification method using the Email Notification Service, by clicking the Enable Email Notification Service in the IMAP connector configuration pane. If you configure both then both the methods would pull the data from the same source and there could be data loss due to conflicts.
- If you are using the IMAP connector for data ingestion and at the field mapping stage you change the default module for data ingestion, which is set as **Alerts** to any other module, for example, **Emails**, then the playbook used for ingesting data remain active and no records will be created. To get the data ingestion to work correctly, do the following:
  Open the `> IMAP > Create Record` playbook and click the **Upsert File Indicators** step. In the Upsert File Indicators step, you need to initialize the IRI value of the module that you want to use for data ingestion instead of

the alert record IRI. To achieve this, click the **emailRecordIRI** field, and use "Dynamic Values" to enter the IRI value of the required module or enter the following Jinja value: `{{vars.steps.Create_Record['@id']}}`.
For example, if you have changed the module used for data ingestion from Alerts to Emails, then to initialize the `emailRecordIRI` variable instead of `alertRecordIRI`, in the **emailRecordIRI** field, enter the `{{vars.steps.Create_Record['@id']}}` value.
Also, note that If you select any module other than the default module, you will require to remap all the fields.

- If you are using the Syslog connector for data ingestion, then you must ensure playbooks used for data ingestion provide input in the CEF format, otherwise you require to modify the playbooks. Since CEF is the most common format for Syslog messages, the default data ingestion playbooks parse the message as CEF. If the Syslog messages from your server are non-CEF complaint, you can modify the "Parse CEF" step in the `> Syslog> Fetch` playbook that is part of the playbooks shipped with the Syslog connector, before you configure Data Ingestion. The Syslog connector also provides "Parse Message" action to parse RFC 3164 and RFC 5424 formatted messages and convert these message formats to CEF format, instead of using the "Parse CEF" step.

Troubleshooting

The Fetch Data screen in case of IMAP displays the`ERROR :: create failed: [ALREADYEXISTS] Folder name ... error

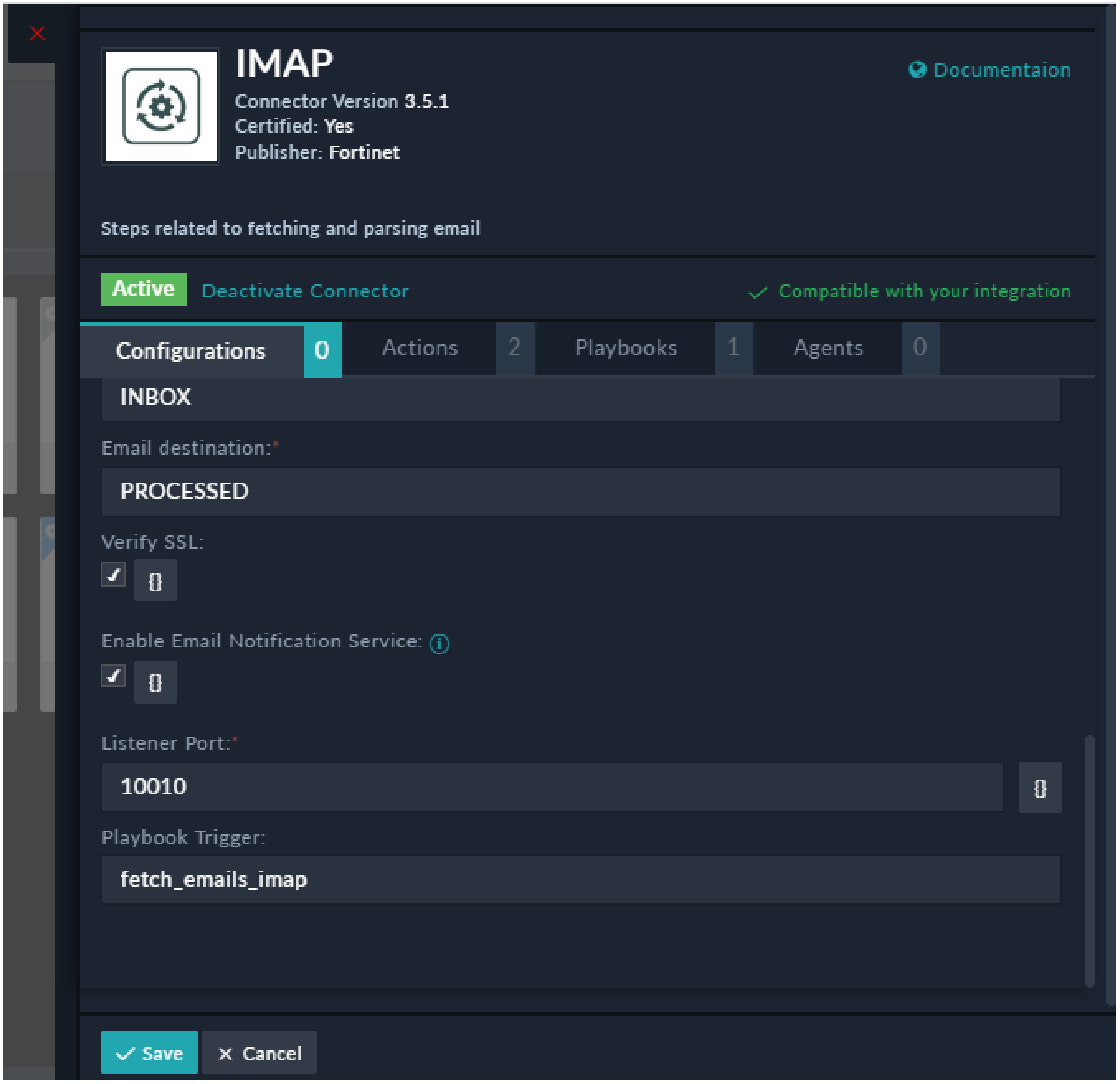
This error occurs when you are running the Data Ingestion Wizard using the IMAP connector. Click **Fetch Data** on the `Fetch Data` screen and an error such as "`> IMAP > Fetch has failed because CS_INTEGRATION-5: Error occured while executing the connector action ERROR:: create failed [ALREADYEXISTS] Folder name conflicts with existing folder name........`" is displayed.

The above error can occur due to the following reasons:

- This issue occurs if the **Email destination** folder specified on server is similar to an existing folder. Mailbox names are case sensitive on creation of folders using the IMAP client, for example, if you specify MyFolder in the in the **Email destination** field and a folder named myfolder already exists on the server, then this error will be displayed on the Fetch Data screen when you are running the Data Ingestion Wizard.
- This error also occurs if the folder that you have specified does not have permission for the IMAP client.

Resolution

- In the IMAP configuration pane, update the folder name in the **Email destination** field and save the configuration:



IMAP
Connector Version 3.5.1
Certified: Yes
Publisher: Fortinet

Steps related to fetching and parsing email

Active Deactivate Connector ✓ Compatible with your integration

Configurations 0 | Actions 2 | Playbooks 1 | Agents 0

INBOX

Email destination:*
PROCESSED

Verify SSL:
✓

Enable Email Notification Service:
✓

Listener Port:*
10010

Playbook Trigger:
fetch_emails_imap

- Ensure that you provide appropriate permissions to the email destination folder for the IMAP client.

Post upgrade to FortiSOAR 6.0.0 or later from version 5.1.0 does not save any data ingestion configurations

If you have upgraded your FortiSOAR version from 5.1.0 to 6.0.0 or later, you will see data ingestion configurations that you had saved in version 5.1.0 are not available in version 6.0.0. You will not see any schedules, ingestion playbook link, or actions button on the `Data Ingestion` page. However, your data ingestion will continue to run. This is because support for fetching data for each configuration of your connector has been introduced in version 6.0.0 and therefore, the older data ingestion configurations and playbooks have been deprecated.

Resolution

Rerun the data ingestion wizard, i.e., reconfigure your data ingestion if required. Note that your previously configured data ingestion will continue to run, however, you will not be able to see the historical data. If you do require to reconfigure the data ingestion, then you must ensure that you stop your previous configured data ingestion, otherwise both the current and the old data ingestion will continue to run.

www.fortinet.com