TruSTAR App for Resilient

App Specification Guide v1.2.0

VERSION CONTROL

#	Document Version	Date	Owner	Document Status	Comments
1	v1.0.0	29-Jun-2018	Hardik Shingala	Draft	First Draft Release
2	v1.1.0	12-Oct-2018	Hardik Shingala	Draft	Added python and 3 compatibility
3	v1.2.0	28-Nov-2018	Hardik Shingala	Draft	Added Proxy Support

Contents

Introduction

This document is intended to provide overall App Specification for the TruSTAR App built for IBM Resilient. It contains details of overall App specification and use cases which will be executed as part of this integration.

Installation Prerequisites

Before installing, verify that your environment meets the following prerequisites:

- Resilient platform is version 30 or later.
- You have a Resilient account to use for the integrations. This can be any account that has the permission to view and modify administrator and customization settings and read and update incidents. You need to know the account username and password.
- You have access to the command line of the Resilient appliance, which hosts the Resilient platform; or to a separate integration server where you will deploy and run the integration. If using a separate integration server, you must install Python version 2.7.10 or later, or version 3.6 or later, and "pip". (The Resilient appliance is preconfigured with a suitable version of Python.)
- You have access to TruSTAR platform and have details like api_key, api_secret and enclave ids for report submission and query.
- Install rc_webserver and rc_cts packages. Click here to download the packages. Use version >= 29.1.0 of these packages.

Install the Python components

The functions package contains Python components that will be called by the Resilient platform to execute the functions during your workflows. These components run in the 'resilient-circuits' integration framework.

The package also includes Resilient customizations that will be imported into the platform later.

Ensure that the environment is up to date,

```
sudo pip install --upgrade pip
sudo pip install --upgrade resilient-circuits
```

To install the package, you must first unzip it then install the package as follows:

```
sudo pip install --upgrade trustar_resilient -<version>.tar.gz
```

Configure the Python components

The 'resilient-circuits' components run as an unprivileged user, typically named `integration`. If you do not already have an `integration` user configured on your appliance, create it now.

Perform the following to configure and run the integration:

1. Use one of the following commands to create or update the resilient-circuits configuration file. Use –c for new environments or –u for existing environments.

   ```
   sudo resilient-circuits config -c
   ```

   or

   ```
   sudo resilient-circuits config -u
   ```

2. Edit the resilient-circuits configuration file (Use sudo if editing using text editor in Linux).
   a. In the [resilient] section, ensure that you provide all the information needed to connect to the Resilient platform.
   b. In the [trustar] section, you will see a field 'queue', which contains the name of the message destination which will be used for this integration. Change the name if you want to use some other message destination else leave it unchanged. (Make sure that the user that you specified in [resilient] section has access to the message destination you specify.
   c. If the integration is running on a machine and is using proxy for internet connectivity, then need to add proxy related configuration in [trustar] section as below:

```
[trustar]


# Name of the message destination.
queue = trustar
# Set the value true if proxy is enabled on the machine where this utility is running.
proxy = false
# Set below property as true if secured proxy is in use.
secure_proxy = false
# URL of proxy server in ip:port(8.8.8.8:1111) format
proxy_url =
# Username of secure proxy
proxy_username = ^proxy_username_for_trustar
# Password of secure proxy
proxy_password = ^proxy_password_for_trustar
```

d. In the [trustar_threat_source] section update following fields:

```
# URL of TruSTAR platform.
url =
# API key of user from TruSTAR platform. Do not change this.
user_api_key = ^api_key_for_trustar_threat_source
# API secret of user from TruSTAR platform. Do not change this.
user_api_secret = ^api_secret_for_trustar_threat_source
# Enclave IDs of user from TruSTAR for searching indicators. Separate values using #comma.
enclave_ids_for_submission =
```

e. In the [trustar_account_n] section change the 'n' with some integer number and update following fields in that:

```
# URL of TruSTAR platform.
url = trustar url
# API key of user from TruSTAR platform.
user_api_key = ^api_key_for_[stanza_name (for e.g. trustar_account_n)]
# API secret of user from TruSTAR platform.
user_api_secret = ^api_secret_for_[stanza_name (for e.g. trustar_account_n)]

# Enclave IDs of user from TruSTAR for submitting report. Separate values using comma.
enclave_ids_for_submission = list of enclave ids
# Enclave IDs of user from TruSTAR from querying on TruSTAR. Separate values using comma.
enclave_ids_for_query = list of enclave ids
# Auto Submission parameter. Possible values - enable|disable
auto_submission = disable
# Enter parameters to submit with report to TruSTAR. Possible values - #Summary|Notes|Breach|Artifacts. Separate values using comma.
submit_data_to_trustar = summary,notes,breach,artifacts
# Incident Types to exclude for report submission to TruSTAR. Separate values using comma.
incident_types_to_exclude = Denial of Service
# List of workspaces for which this TruSTAR account will be used.
workspace = list of workspaces
# TAG to assign to the report submitted to TruSTAR
tag =
```

f. You can add other TruSTAR accounts for different workspaces. Append a new section named [trustar_account_n] with some other value of ‘n’ and repeat step 2.d.

g. If you want to use Threat service feature of the integration, you need to add following sections in app.config file.(Ignore if you have already installed rc-cts and rc-webserver community apps of resilient and added configuration for both in app.config file)

    i. Add [webserver] section in app.config file and in that section add following fields:

```
server = Host IP of server where resilient circuits is running (Default is local host)
port = Post on which this will run (Default is 9000)
secure = 1|0 (1 for https, 0 for http. Default is 0)
cafile = certificate file. (Needed only when secure = 1)
```

    j. Add [custom_threat_service] section in app.config file and in that section add following fields: (Remove # from starting of field if you want to make any changes.)

```
urlbase=/cts (SHOULD NOT CHANGE)
#first_retry_secs= 5
#later_retry_secs=60
#max_retries=60
#cache_size=10000
#cache_ttl=600000
```

    k. If you want to change value of urlbase in the above section, update the same value while you register threat service and in place of cts in url for threat service.

h. After doing changes in the config file run following command:

```
sudo res-keyring
```

i. Here you will need to provide details like API key and API secret.

j. User can also use this method to store their Resilient platform.

    ii. In [resilient] stanza of app.config file, replace value of password with “^password”.

    iii. Then run this command and it will ask you to enter the password value.

3. The package contains rules, message destination and function definitions that you can use in workflows.

   Deploy these customizations to the Resilient platform with the following command:

   ```
   resilient-circuits customize
   ```

   Answer the prompts to deploy functions, message destinations, workflows and rules.

Run the integration framework

1. Steps for Linux

a. Create a service file using following command.

```
sudo vi /etc/systemd/system/resilient_circuits.service
```

b. Add following content in that .service file:

```
[Unit]
Description=Resilient-Circuits Service
After=resilient.service
Requires=resilient.service

[Service]
Type = simple
User = root
WorkingDirectory = /root
ExecStartPre = /usr/bin/resutil threatserviceedit -name “TruSTAR” \
                                        -resturl “{http|https}://127.0.0.1:9000/cts/trustar”
ExecStart = /usr/local/bin/resilient-circuits run -r
Restart = always
TimeoutSec = 100

[Install]
WantedBy = multi-user.target
```

Change locations in the file as per the environment.

c. Ensure that the service unit file is correctly permissioned:

```
sudo chmod 664 /etc/systemd/system/resilient_circuits.service
```

d. Use the systemctl command to manually enable or disable the service:

```
sudo systemctl [enable|disable] resilient_circuits
```

e. Use the systemctl command to manually start, stop, restart and return status on the service:

```
sudo systemctl [start|stop|restart|status] resilient_circuits
```

f. Log files for systemd and the resilient-circuits service can be viewed through the journalctl command:

```
sudo journalctl -u resilient_circuits --since "2 hours ago"
```

2. Steps for Windows

a. Run the following command from command prompt.

```
resilient-circuits run -r
```

b. Run the following command where your resilient platform is installed.

```
sudo resutil threatserviceedit -name "TruSTAR" -resturl "{url}/cts/trustar"
- In place of {url}, add value in this format:
{http|https}://host_ip{port_you_added_in_config_file}
```

Business use case

TruSTAR app provides the end-user a holistic view for SOC Admin to troubleshoot security offenses and improve overall security posture of the organization. Integration with Resilient to allow Resilient users to insert all the incidents being created, to TruSTAR and also empower them to enrich incidents and events in the Resilient.

Following are use cases are addressed in the TruSTAR app for Resilient.

1. Assign an Incident to a particular workspace.
2. Submit Incident data to TruSTAR as reports (GUI Manual Action or Automatic Action)
3. Update submitted report on TruSTAR whenever a new artifact is added, or status of the incident is changed. (GUI Manual Action or Automatic Action)
4. Fetch correlated indicators from TruSTAR whenever a new incident is added, or currently submitted incident is updated (GUI Manual Action or Automatic Action)
5. Add notes added in Resilient Incident to TruSTAR report whenever an Incident status is changed to Close. (GUI Manual Action or Automatic Action)
6. Add resolution and resolution summary of a closed resilient incident to its corresponding report in TruSTAR. (GUI Manual Action or Automatic Action)
7. Delete Report of a deleted incident from TruSTAR. (Automatic Action)
8. Whitelist and Undo Whitelist indicators in TruSTAR. (GUI Manual Action)
9. Workflow functions to perform following actions:
    a. Submit or Update Incident data to TruSTAR
    b. Get Priority Score of an Indicator from TruSTAR
    c. Get Correlated Indicators from TruSTAR for a particular Incident
    d. Delete report for a particular incident from TruSTAR
    e. Whitelist an indicator in TruSTAR
    f. Undo Whitelist an indicator in TruSTAR
10. Threat Service

Here are the details on overall use cases addressed in TruSTAR app for Resilient.

1. Submit Incident to different TruSTAR accounts or enclaves based on workspace.

    - Using this feature, incidents from different workspaces can be submitted to different TruSTAR accounts or enclaves.
    - Mapping of workspaces with TruSTAR accounts/enclaves need to be done via app.config file sections.
    - To assign an incident to a particular workspace perform following steps.
        - Navigate to Customization Settings

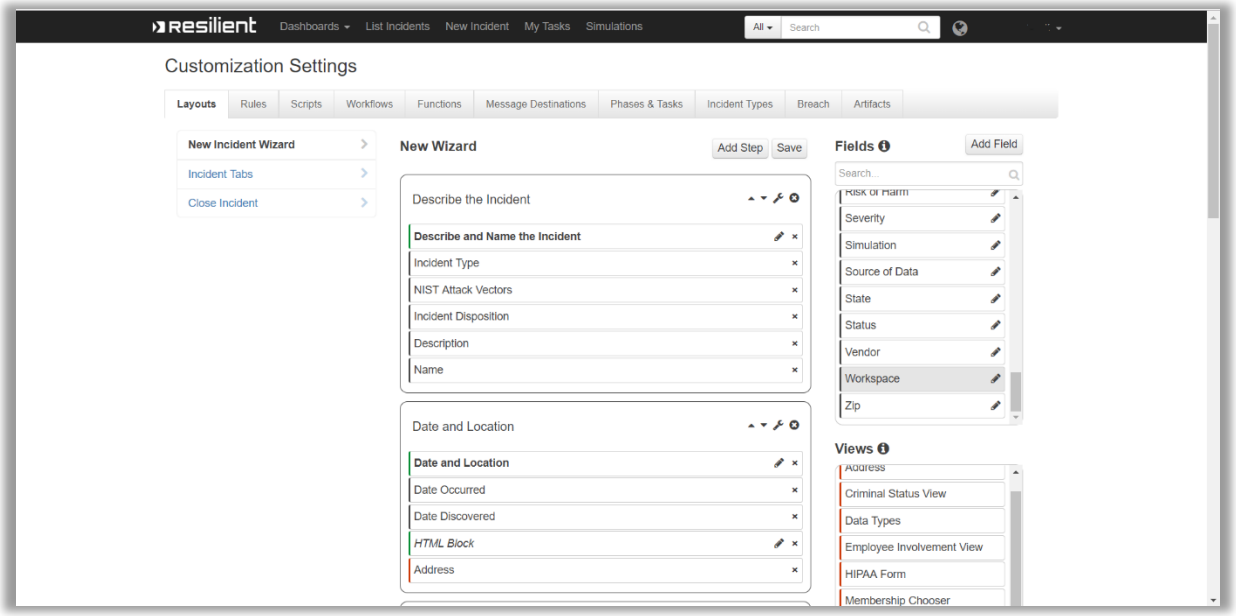

Figure 1 Navigate to Customization Settings

- Navigate to Layouts tab under Customization Settings.

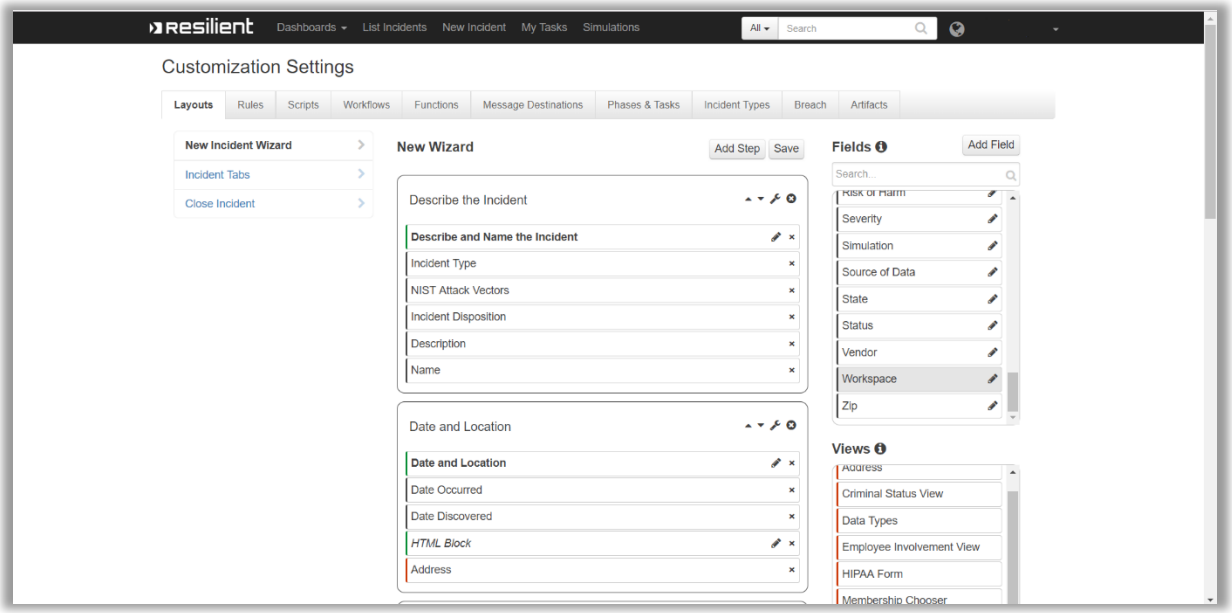

Figure 2 Navigate to Layouts under Customization Settings

- Under New wizard, add Workspace field from Fields to one of the blocks.

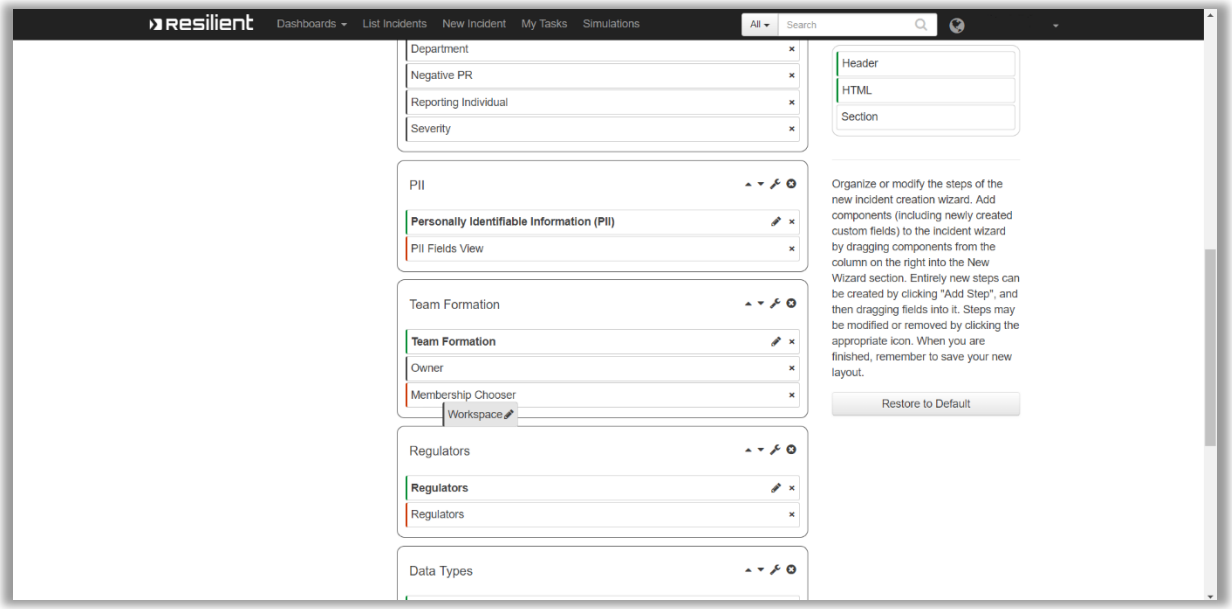

Figure 3 Place workspace field at one wizard

- Click on the save button to save the changes.

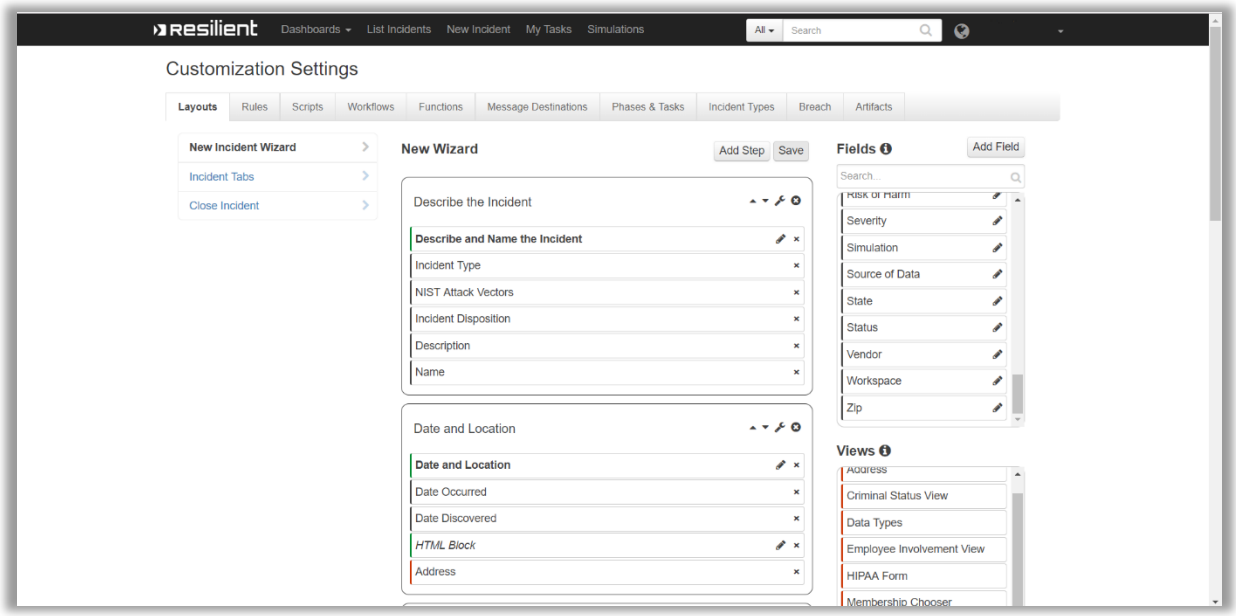

Figure 4 Save the changes

- Now you can select workspace whenever you create a new incident.

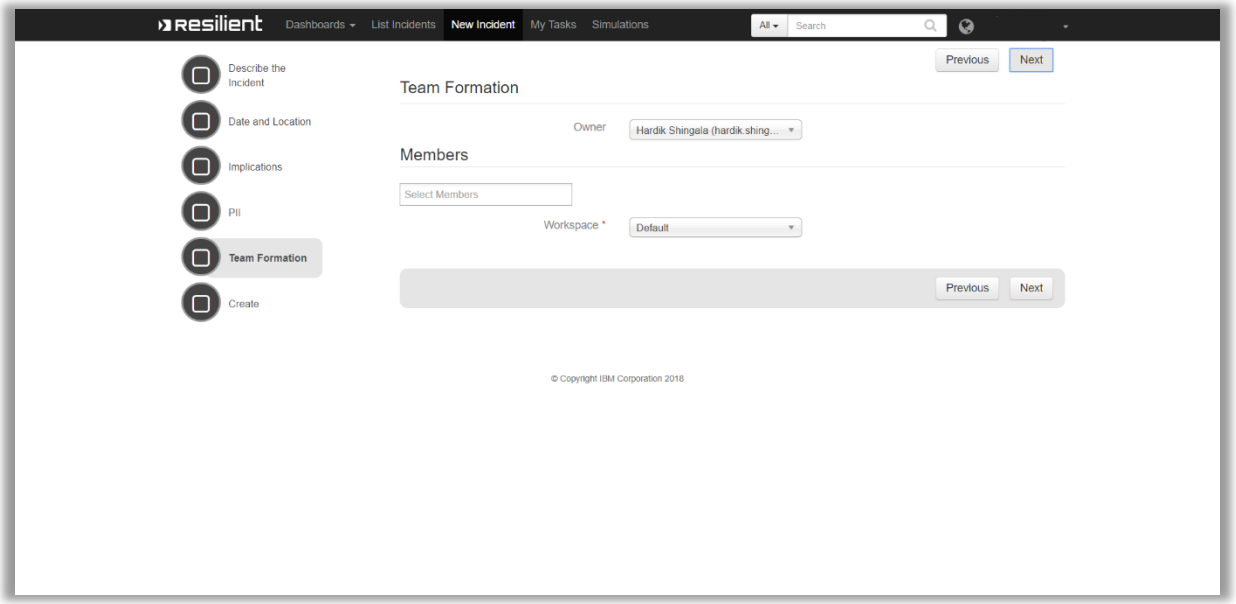

Figure 5 Select workspace for new incident

2. Submit Incident data to TruSTAR as reports (GUI Manual Action or Automatic Action)

- User can submit an incident as a report in TruSTAR
- Deeplink of the submitted report will be added as a note in the incident.
- Report can be submitted automatically or using manual action.
- If user has enabled auto_submission in app.config, then report will be submitted automatically.
- If user has disabled auto_submission in app.config, then he/she need to perform "Send To TruSTAR" manual action to submit report.
- Steps for Manual Action:
    - Navigate to an Incident listed under "List Incidents" menu.
    - On that incident find "Actions" dropdown tab at upper right corner.
    - Click "Send To TruSTAR" under that dropdown.

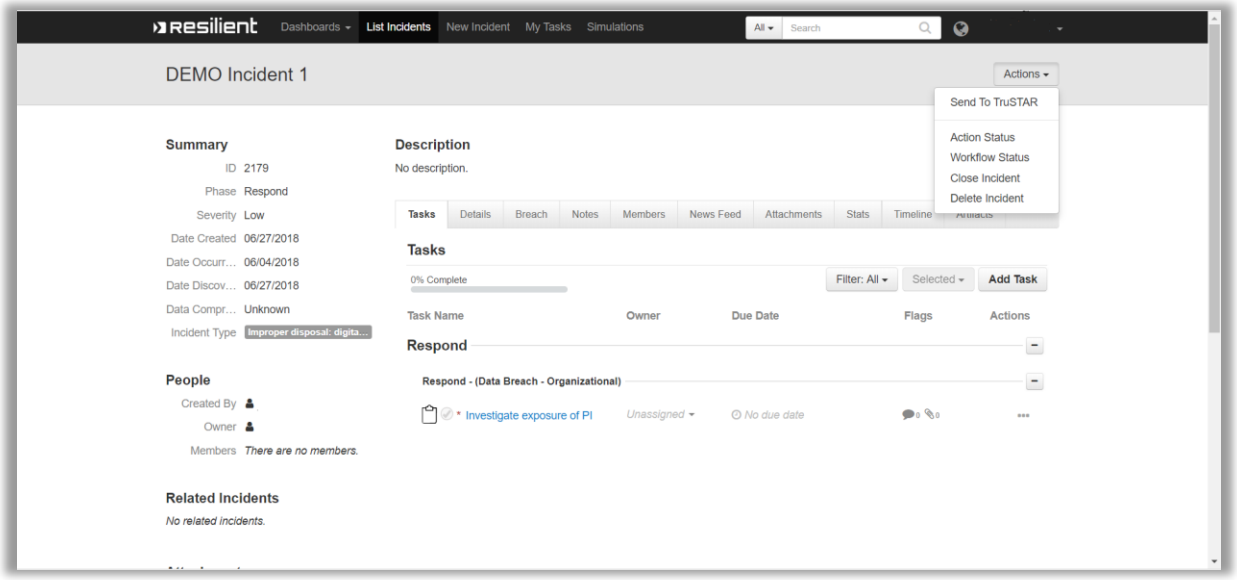

Figure 6 Send Incident to TruSTAR

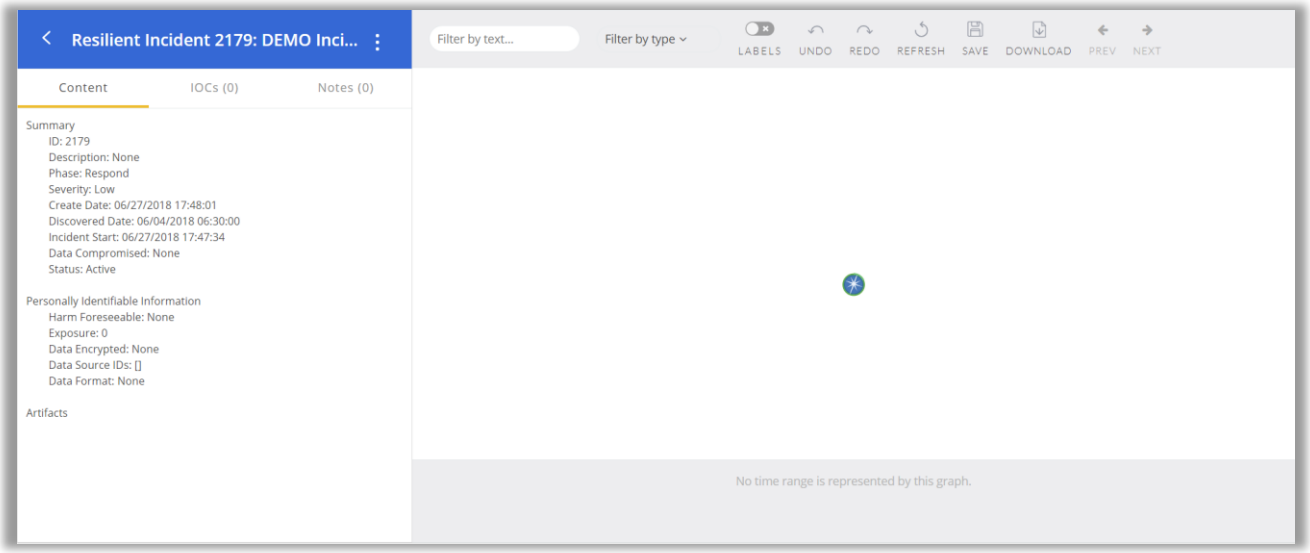

Figure 7 Report submitted to TruSTAR

3. Update submitted report on TruSTAR whenever a new artifact is added, or status of the incident is changed. (GUI Manual Action or Automatic Action)

    - Using this functionality, whenever a new artifact is added in Resilient incident and auto submission is enabled, corresponding report of that incident in TruSTAR will added.
    - If auto submission is disabled, user need to perform “Send To TruSTAR” manual action shown in Figure 1.
    - Steps are the same which you have followed while submitting report.

4. Fetch correlated indicators from TruSTAR whenever a new incident is added, or currently submitted incident is updated (GUI Manual Action or Automatic Action)

    - Using this functionality, whenever an incident will be submitted or updated to TruSTAR automatically or manually, correlated indicators for that report will be fetched and each of the correlated indicator will be added as an artifact, as well as list of all correlated indicators will be added as note with deeplink of each indicator in incident.

5. Add notes added in Resilient Incident to TruSTAR report whenever an Incident status is changed to Close. (GUI Manual Action or Automatic Action)

    - This feature will add all notes which are added in an incident to its corresponding report whenever that incident is closed.
    - If auto submission is enabled, report will be updated automatically.
    - If auto submission is disabled, user need to perform “Send To TruSTAR” manual action to reflect the changes in its corresponding TruSTAR report.

6. Add resolution and resolution summary of a closed resilient incident to its corresponding report in TruSTAR. (GUI Manual Action or Automatic Action)

    - This feature will add resolution and resolution summary of a closed incident, to its corresponding report in TruSTAR.
    - If auto submission is enabled, report will be updated automatically.
    - If auto submission is disabled, user need to perform “Send To TruSTAR” manual action to reflect the changes in its corresponding TruSTAR report.

7. Delete Report of a deleted incident from TruSTAR. (Automatic Action)

    - This feature will delete report of a deleted incident from TruSTAR if auto submission is enabled.
    - If auto submission is disabled, user need to delete report manually from TruSTAR.

8. Whitelist and Undo Whitelist indicators in TruSTAR. (GUI Manual Action)

- This feature lets resilient user to whitelist an artifact in TruSTAR or remove an already whitelisted artifact from TruSTAR
- Steps:
  - Navigate to an Incident listed under “List Incidents” menu.
  - Under that incident, navigate to Artifacts tab.
  - Under that tab click on button showed in below screenshot to perform these actions.

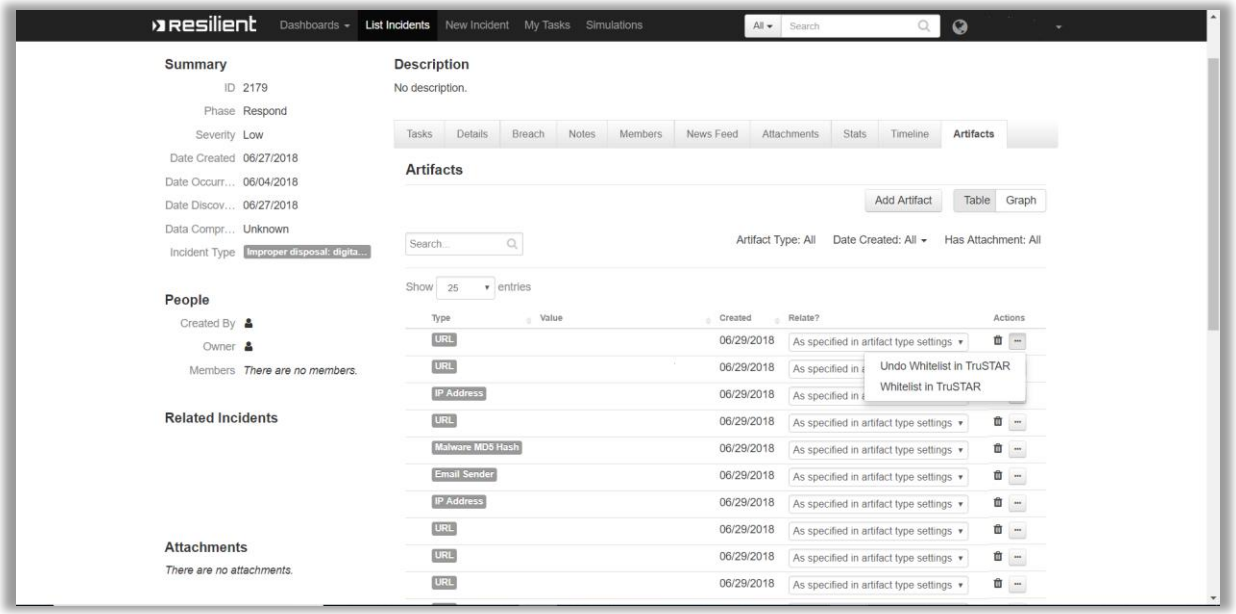

Figure 8 Whitelist | Undo whitelist artifact in TruSTAR

9. Workflow functions to perform following actions:

    a. Submit or Update Incident data to TruSTAR
        i. Input: Incident ID
        ii. Pre-process script

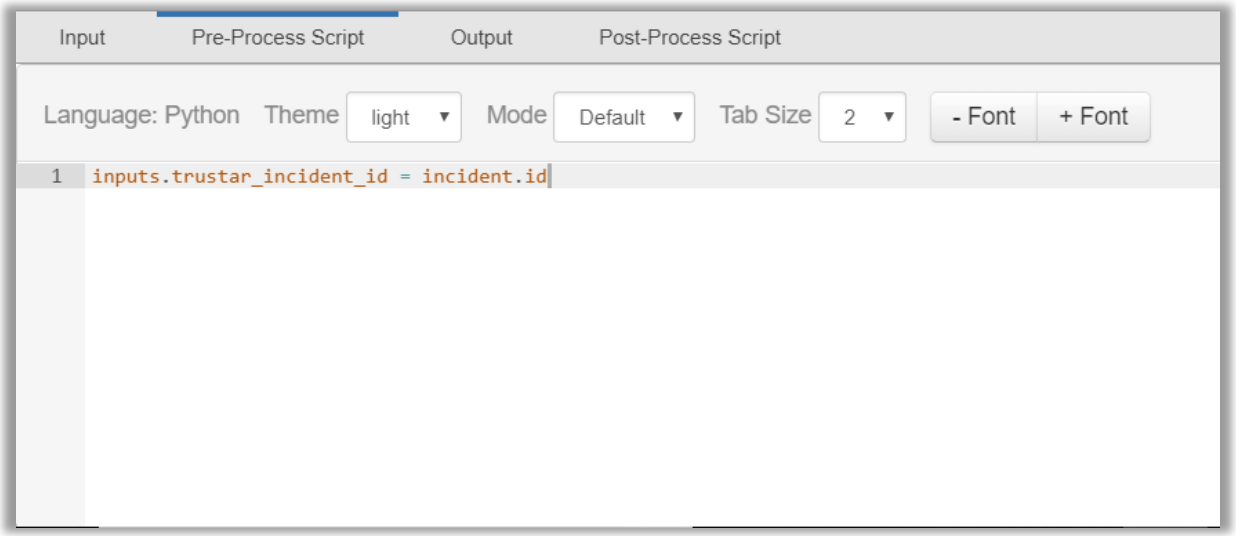

Figure 9 Pre-process script having trustar_incident_id as input

        iii. Output: Report submitted to TruSTAR, in json format
        iv. Payload: JSON object with following content

Key	Value
reportBody	Incident content submitted to TruSTAR
updated	Time of last report update
externalUrl	None
created	Time when report was created
distributionType	ENCLAVE
title	Report Title
timeBegan	Time that the incident began
id	Report ID
enclaveIds	ID of enclaves in which report is submitted
externalTrackingId	External ID of report

b. Get Priority Score of an Indicator from TruSTAR
   i. Input: Indicator value
   ii. Pre-process script

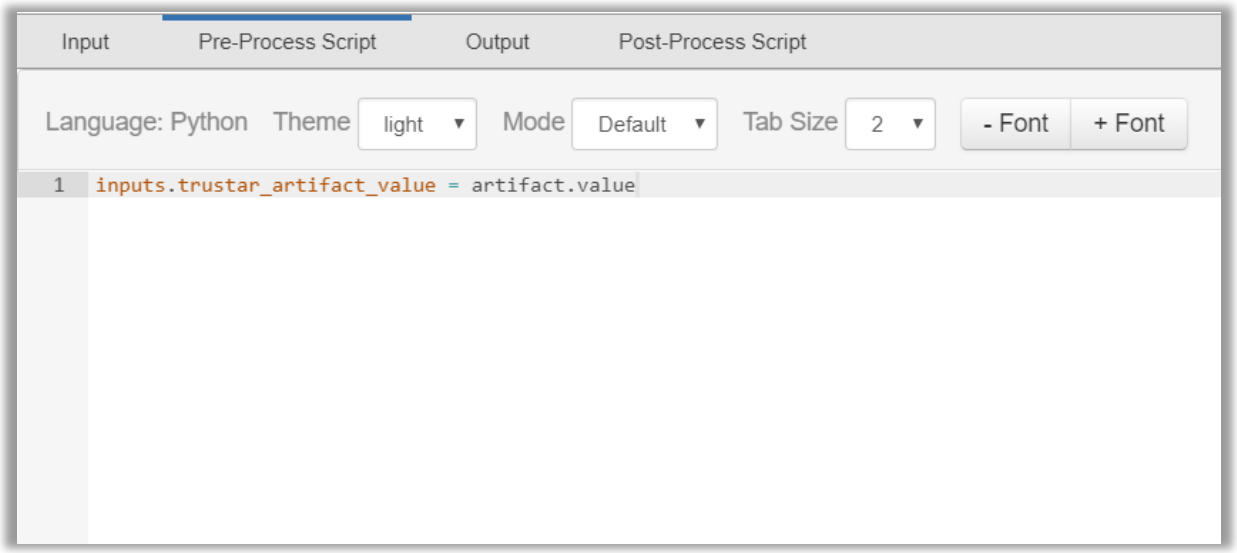

Figure 10 Pre-process script having trustar_artifact_value as input

   iii. Output: Priority Value of the provided artifact in json format.
   iv. Payload: JSON object with following content

Key	Value
priorityLevel	Priority level of that indicator in TruSTAR (LOW, MEDIUM, HIGH, NOT_FOUND)

c. Get Correlated Indicators from TruSTAR for a particular Incident
   i. Input: Incident ID
   ii. Pre-process script: Same as figure. 9
   iii. Output: List of corelated indicators in json format
   iv. Payload: JSON object with following content

Key	Value

indicators	List of correlated indicators.
	E.g.:
	[
	{
	‘type’: “MALWARE”,
	‘value’: “WANNACRY”
	}
	]

d. Delete report for a particular incident from TruSTAR
    i. Input: Incident ID
    ii. Pre-process script: Same as figure. 9
    iii. Output: Status in json format, done or error.
    iv. Payload: JSON object with following content

Key	Value
status	Done

e. Whitelist an indicator in TruSTAR
    i. Input: Indicator value and incident ID
    ii. Pre-process script

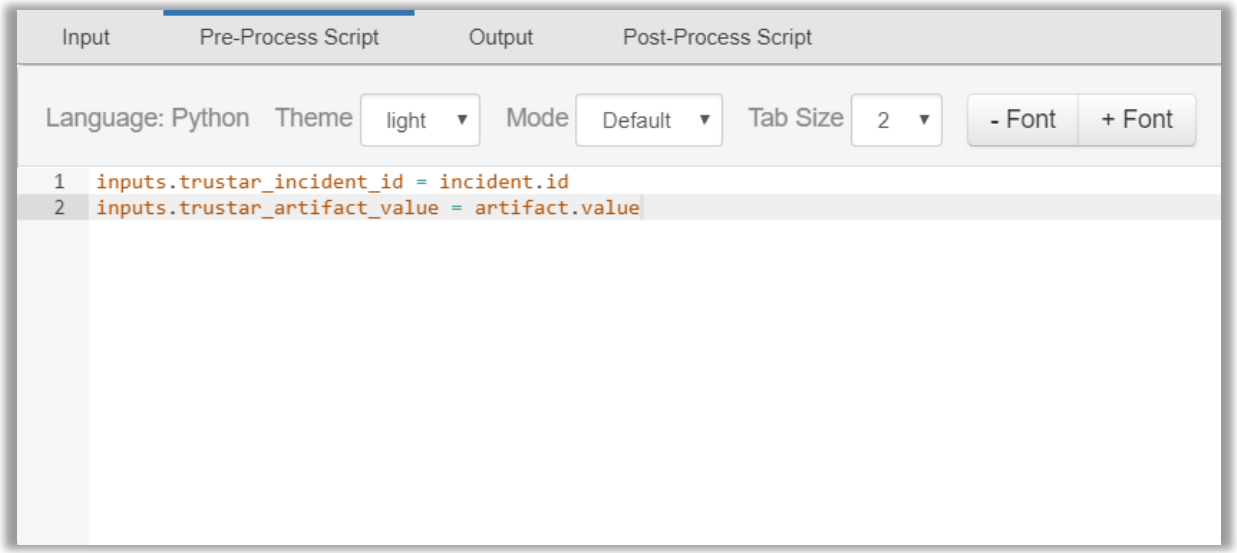

Figure 11 Pre-process script having trustar_artifact_value and trustar_incident_id as inputs

    iii. Output: List of indicators whitelisted in TruSTAR
    iv. Payload: JSON object with following content

Key	Value

indicators	List of indicators.
	E.g.:
	[
	{
	‘type’: “MALWARE”,
	‘value’: “WANNACRY”
	}
	]

f. Undo Whitelist an indicator in TruSTAR
    i. Input: Indicator value, type and Incident ID
    ii. Pre-process script

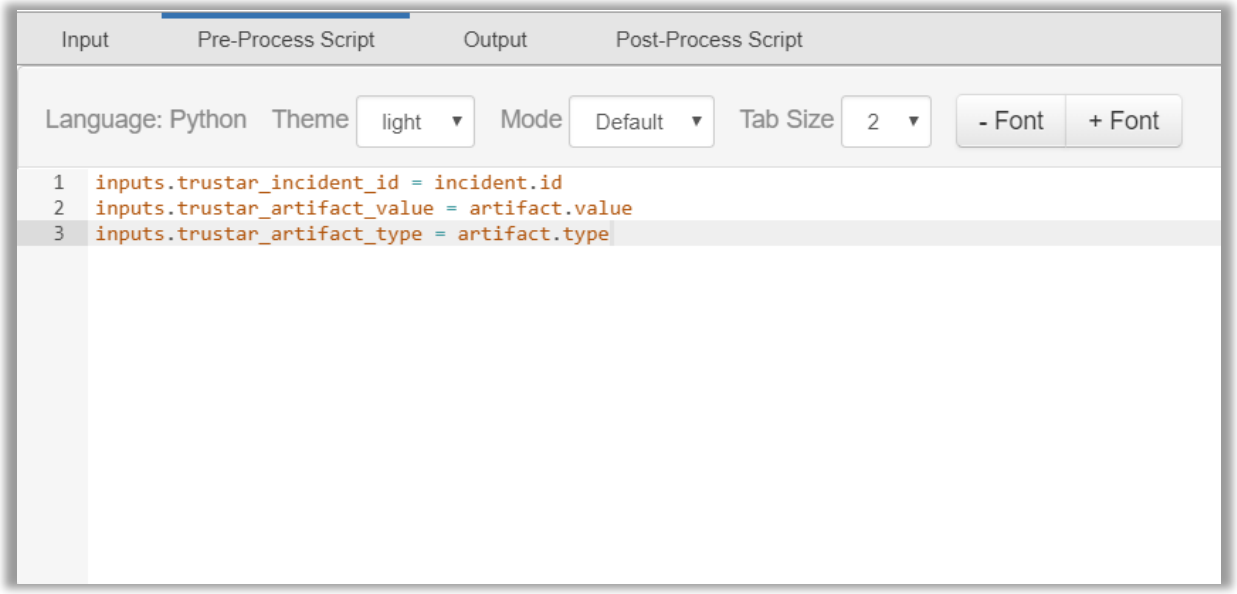

Figure 11 Pre-process script having trustar_artifact_value, trustar_artifact_type and trustar_incident_id as inputs

    iii. Output: Status in json format, done or error.
    iv. Payload: JSON object with following content

Key	Value
status	Done

Note

- If any error occurs during execution of any of the above functions than the output of that function will be a JSON object with 'error' key and error message as value.

Post-process script

- Result of any function is accessible in post-process script using "**results"** variable.
- See below example to understand how you can fetch results.

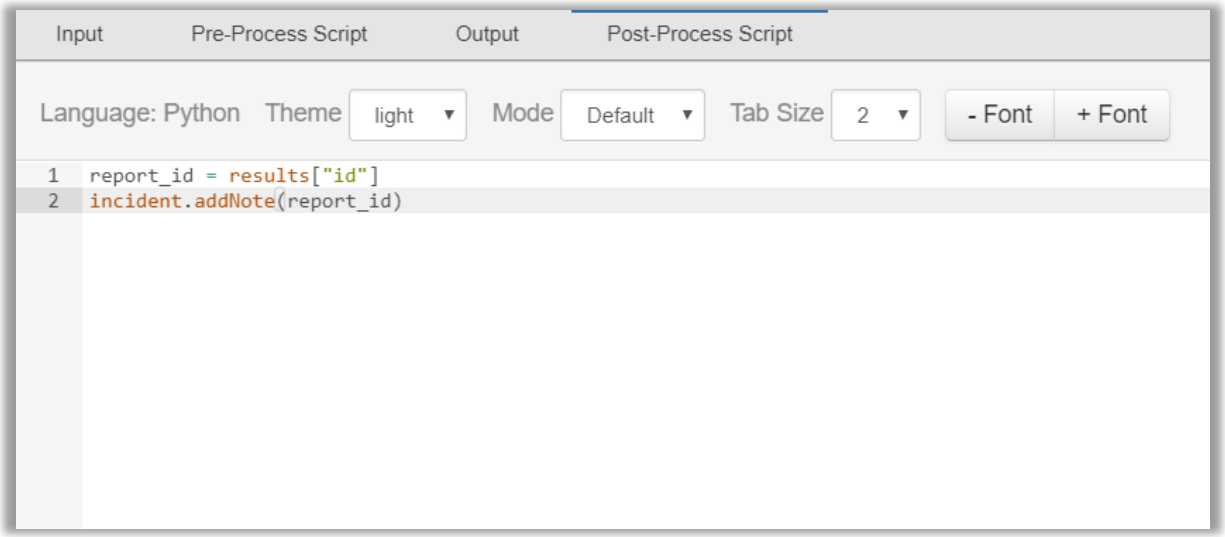

Figure 12 Post-process script for Submit Report to TruSTAR function.

- Here, ID of submitted report is fetched from the "Submit report to TruSTAR" function result and added as note in incident.
- Using the same approach, you can fetch results of other functions and use them in your business logic.

10. Threat Service

- Whenever a new artifact is added in any incident, this feature will fetch its priority score from the TruSTAR and update that in artifact's hits section.
- To enable or disable this feature perform following steps:

    1. Navigate to Administrator Settings.

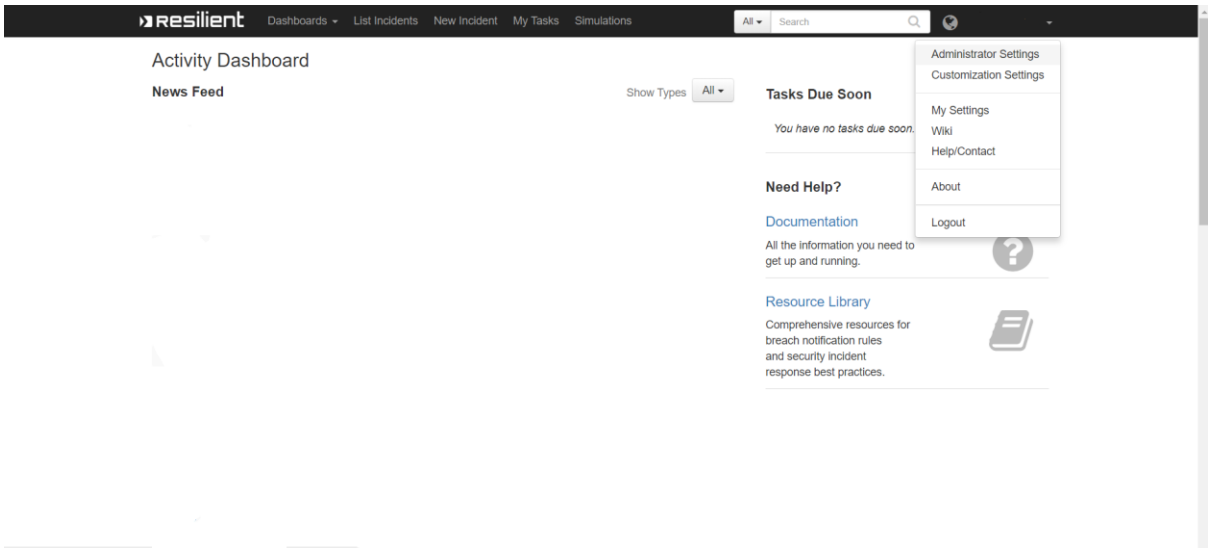

Figure 13 Navigate to Administrator Settings

    2. Navigate to Threat Sources tab under Administrator Settings

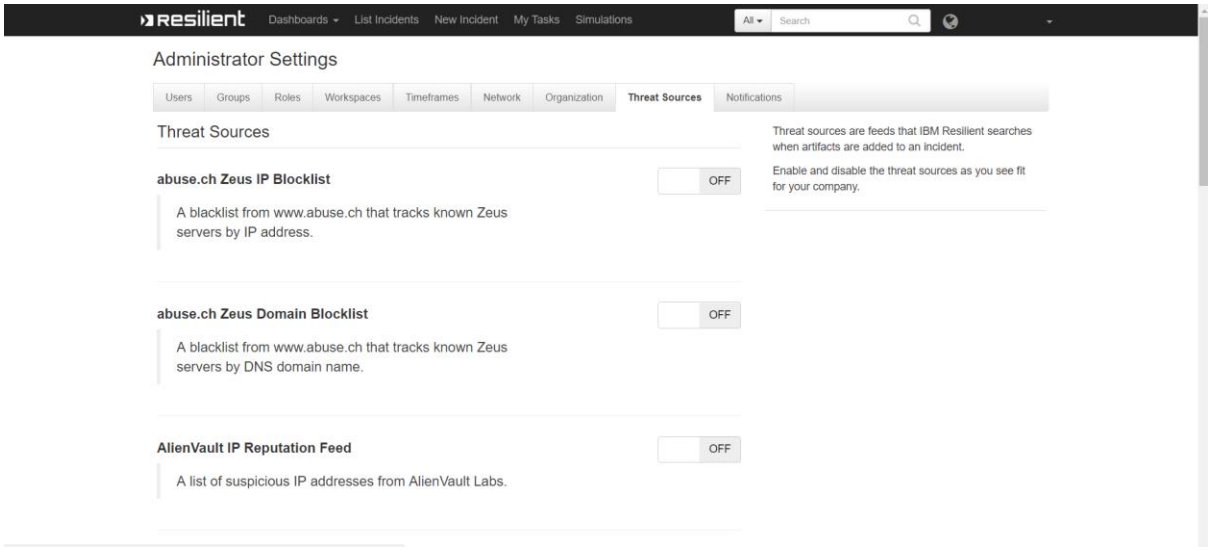

Figure 14 Navigate to Threat Sources

3. Find “TruSTAR” threat source. You can turn its status from ON|OFF from here.

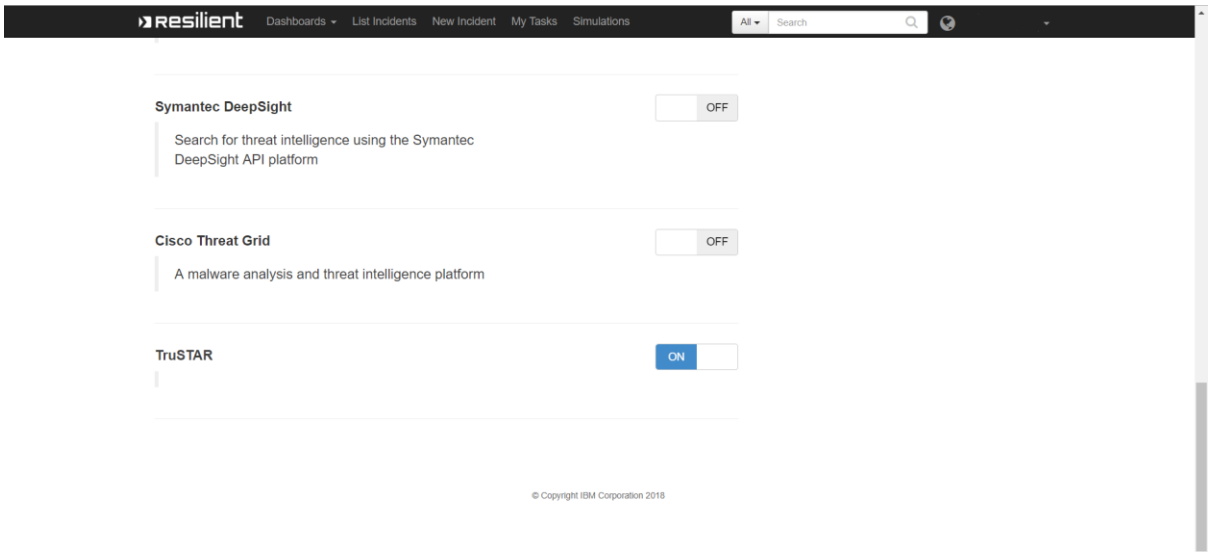

Figure 15 TruSTAR threat source

Limitations

Below are the known limitations of this integration:

1. It will take around 10 – 15 seconds to fetch the data from TruSTAR and reflect those data in Resilient.
2. User needs to refresh the page to view the data enriched in Resilient.
3. Some indicators of URL type returned from the TruSTAR are not accepted by Resilient.
   Solution: We have added an artifact type “URL String”, which will be assigned as type to those indicators.
4. Proxy support is not provided with this integration.