

ThreatQuotient



Request Tracker Connector User Guide

Version 2.1.0

October 18, 2023

ThreatQuotient

20130 Lakeview Center Plaza Suite 400
Ashburn, VA 20147

 **ThreatQ Supported**

Support

Email: support@threatq.com

Web: support.threatq.com

Phone: 703.574.9893

Contents

Warning and Disclaimer	3
Support	4
Integration Details.....	5
Introduction	6
Prerequisites	8
Time Zone	8
Integration Dependencies	9
Installation.....	10
Creating a Python 3.6 Virtual Environment	10
Installing the Connector.....	11
Configuration	14
Configuring the Extension	15
Usage.....	16
Command Line Arguments.....	16
CRON	18
Supported Custom Fields	19
Enabling Custom Fields.....	19
Upgrading the Extension	21
Change Log	22

Warning and Disclaimer

ThreatQuotient, Inc. provides this document “as is”, without representation or warranty of any kind, express or implied, including without limitation any warranty concerning the accuracy, adequacy, or completeness of such information contained herein. ThreatQuotient, Inc. does not assume responsibility for the use or inability to use the software product as a result of providing this information.

Copyright © 2023 ThreatQuotient, Inc.

All rights reserved. This document and the software product it describes are licensed for use under a software license agreement. Reproduction or printing of this document is permitted in accordance with the license agreement.

Support

This integration is designated as **ThreatQ Supported**.

Support Email: support@threatq.com

Support Web: <https://support.threatq.com>

Support Phone: 703.574.9893

Integrations/apps/add-ons designated as **ThreatQ Supported** are fully supported by ThreatQuotient's Customer Support team.

ThreatQuotient strives to ensure all ThreatQ Supported integrations will work with the current version of ThreatQuotient software at the time of initial publishing. This applies for both Hosted instance and Non-Hosted instance customers.



ThreatQuotient does not provide support or maintenance for integrations, apps, or add-ons published by any party other than ThreatQuotient, including third-party developers.

Integration Details

ThreatQuotient provides the following details for this integration:

Current Integration Version 2.1.0

**Compatible with ThreatQ
Versions** $\geq 3.6.0$

Python Version 3.6

Support Tier ThreatQ Supported

Introduction

The Request Tracker Custom Connector ensures that Request Tracker tickets that have been created in ThreatQ are continuously synced with Request Tracker. It polls ThreatQ's API and checks for any updated/touched Request Tracker tickets (events). If it detects an updated Request Tracker ticket, it will find that ticket in Request Tracker and update it with the new information. This section of the overall integration is installed as any other ThreatQ integration and will be further configured via the Incoming Feeds page in the ThreatQ user interface.

Request Tracker Extension

The Request Tracker Extension is designed to asynchronously work with Request Tracker to keep ThreatQ and Request Tracker up to date with each other.

When this extension is installed, it will add the following new 'objects' to your Request Tracker instance:

- 3 New Custom Fields
 - ThreatQ Link
 - ThreatQ Attributes
 - ThreatQ Indicators
- 1 New ScripAction
 - Sync ticket with ThreatQ
- 1 New Scrip
 - Automatically Sync Ticket with ThreatQ
- 2 New Context Menus
 - Ticket Page > ThreatQ > Sync Ticket
 - Home Page > Admin > Tools > Edit ThreatQ Config



These menus will only be viewable by Super Users (admins).

The new custom fields can be applied to Queues, so when you create a ticket in a specified Queue, these fields will be viewable (and some editable). The ThreatQ Link and ThreatQ Indicators fields are not editable and will be populated by ThreatQ when the ticket is synced. The ThreatQ Attributes field is editable and allows you to add attributes to the ThreatQ event using this field.

The new ScripAction will be invoked by a Scrip (rule) when a transaction is made to a ticket. Each transaction has a type. By default, the ticket will be fully synced. However, if the transaction type is part of the following, only certain actions will get invoked. Here is a list of the transaction types and what will happen when invoked:

- Comment/Correspond
 - Syncs comments
- Create
 - Creates the ticket in ThreatQ and syncs all details



A five second delay will be added to this command to ensure the ticket is created.

- AddLink
 - Syncs related tickets
- CustomField
 - Syncs all Custom Fields and attributes/metadata
- Status
 - Syncs the status (Rejects, Resolved, or Queue Name if not the previous two)
- Other
 - Full sync of all fields

In the new context menus, the main menu will be Edit ThreatQ Config. This is where you will edit the configuration that you initially placed in `/opt/rt4/etc/RT_SiteConfig.pm`.

Password fields must be edited from `/opt/rt4/etc/RT_SiteConfig.pm`, as they will not be editable from the configuration page. Once a field is saved (excluding password fields), it will be transferred to the ThreatQ database entry and will not need to be edited from the `/opt/rt4/etc/RT_SiteConfig.pm`. It may then be removed from the SiteConfig. Request Tracker does not have any paradigm for an indicator, so there is no way to transfer indicator attributes from ThreatQ to Request Tracker. Instead, there is a ThreatQ Indicators custom field which will show the related indicators to the ThreatQ event, as well as a link to that indicator in ThreatQ. This will enable the user to easily pivot back and forth between Request Tracker and ThreatQ.

The integration also supports the following Custom Fields (not automatically created by ThreatQ):

- **Tags:** Multiline field
- **External Ticket Number:** Single line field
- **Systems Re-Imaged:** Multiline field
- **Threat TTP:** Multiline field
- **IP:** Multiline field
 - Synced as related 'IP Address' Indicator
- **FileNames:** Multiline field
 - Synced as related 'Filename' Indicator
- **HASH:** Multiline field
 - Synced as related 'MD5', 'SHA-1', or 'SHA-256' Indicator
- **Domains:** Multiline field
 - Synced as related 'FQDN' Indicator
- **Sender:** Single line field
 - Synced as related 'Email Address' Indicator
- **URL:** Single line field
 - Synced as related 'URL' Indicator
- **Threat Actors:** Multiline field
 - Synced as related Adversary
- **Threat Campaign:** Single line field
 - Synced as related Event (type: Campaign)



Request Tracker's native language is PERL, so every extension must be coded in PERL. We have worked around this by installing a python script with the extension. When an action is executed, it will execute a python script to handle most of the work.

Prerequisites

Review the following requirements before attempting to install the connector.

Time Zone

You should ensure all ThreatQ devices are set to the correct time, time zone, and date (UTC is recommended), and using a clock source available to all.

To identify which time zone is closest to your present location, use the `timedatectl` command with the `list-timezones` command line option.


For example, enter the following command to list all available time zones in Europe:

```
timedatectl list-timezones | grep Europe
Europe/Amsterdam
Europe/Athens
Europe/Belgrade
Europe/Berlin
```


Enter the following command, as root, to change the time zone to UTC:

```
timedatectl set-timezone UTC
```


Integration Dependencies

 The integration must be installed in a python 3.6 environment.

The following is a list of required dependencies for the integration. These dependencies are downloaded and installed during the installation process. If you are an Air Gapped Data Sync (AGDS) user, or run an instance that cannot connect to network services outside of your infrastructure, you will need to download and install these dependencies separately as the integration will not be able to download them during the install process.

 Items listed in bold are pinned to a specific version. In these cases, you should download the version specified to ensure proper function of the integration.

DEPENDENCY	VERSION	NOTES
threatqsdk	>=1.6.0	N/A
threatqcc	>=1.3.0	N/A
python-dateutil	N/A	N/A
pytz	N/A	N/A
setuptools	N/A	N/A
six	N/A	N/A

Installation

The following provides you with steps on installing a Python 3 Virtual Environment and installing the connector.

Creating a Python 3.6 Virtual Environment

Run the following commands to create the virtual environment:

```
mkdir /opt/tqvenv/  
sudo yum install -y python36 python36-libs python36-devel python36-pip  
python3.6 -m venv /opt/tqvenv/<environment_name>  
source /opt/tqvenv/<environment_name>/bin/activate  
pip install --upgrade pip  
pip install threatqsdk threatqcc setuptools==59.6.0
```

Proceed to [Installing the Connector](#).

Installing the Connector

⚠ Upgrading Users - Review the [Change Log](#) for updates to configuration parameters before updating. If there are changes to the configuration file (new/removed parameters), you must first delete the previous version's configuration file before proceeding with the install steps listed below. Failure to delete the previous configuration file will result in the connector failing.

1. Navigate to the ThreatQ Marketplace and download the .whl file for the integration.
2. Activate the virtual environment if you haven't already:

```
source /opt/tqenv/<environment_name>/bin/activate
```

3. Transfer the whl file to the /tmp directory on your ThreatQ instance.
4. Install the connector on your ThreatQ instance:

```
pip install /tmp/tqRequestTracker-<version>-py3-none-any.whl
```



A driver called tqRequestTracker will be installed. After installing, a script stub will appear in /opt/tqenv/<environment_name>/bin/tqRequestTracker.

5. Once the application has been installed, a directory structure must be created for all configuration, logs and files, using the `mkdir -p` command. Use the commands below to create the required directories:

```
mkdir -p /etc/tq_labs/
mkdir -p /var/log/tq_labs/
```

6. Perform an initial run using the following command:

```
/opt/tqenv/<environment_name>/bin/tqRequestTracker -ll /var/log/tq_labs/ -c /etc/tq_labs/ -v3
```

7. Enter the following parameters when prompted:

PARAMETER	DESCRIPTION
ThreatQ Host	This is the host of the ThreatQ instance, either the IP Address or Hostname as resolvable by ThreatQ.
ThreatQ Client ID	This is the OAuth id that can be found at Settings Gear → User Management → API details within the user's details.

PARAMETER	DESCRIPTION
ThreatQ Username	This is the Email Address of the user in the ThreatQ System for integrations.
ThreatQ Password	The password for the above ThreatQ account.
Status	This is the default status for objects that are created by this Integration.

Example Output

```
/opt/tqenv/<environment_name>/bin/tqRequestTracker -ll /var/log/tq_labs/
-c /etc/tq_labs/ -v3
ThreatQ Host: <ThreatQ Host IP or Hostname>
ThreatQ Client ID: <ClientID>
ThreatQ Username: <EMAIL ADDRESS>
ThreatQ Password: <PASSWORD>
Status: Review
Connector configured. Set information in UI
```

- SSH into your Request Tracker instance as root (or sudo yourself).
- SCP/Transfer the threatq-request-tracker-extension-1.0.2.tar.gz file into your Request Tracker instance.
- Extract the compressed directory:

```
tar -xzf threatq-request-tracker-extension-1.0.2.tar.gz
```

- Navigate to the extracted directory:

```
cd threatq-request-tracker-extension-1.0.2
```

- Compile a Makefile from PERL Markfile:

```
perl Makefile.PL
```

- Compile the Makefile:

```
make
```

- Install the extension using Makefile:

```
make install
```

- Initialize the database using Makefile.



This step can be skipped if you are upgrading the integration.

```
Make initdb
```



You will be prompted for your password three times. This is normal behavior.

16. Add your `/opt/rt4/etc/RT_SiteConfig.pm` file with the initial ThreatQ configuration:

```
Plugin('RT::Extension::ThreatQ');
Set($ThreatQ_User, '<YOUR THREATQ USERNAME>');
Set($ThreatQ_Password, '<YOUR THREATQ PASSWORD>');
Set($ThreatQ_CID, '<YOUR THREATQ CID>');
Set($RT_User_For_ThreatQ, '<THE RT USER TO USE WITH THREATQ>');
Set($RT_Password_For_ThreatQ, '<THE RT PASSWORD TO USE WITH THREATQ>');
Set($RT_Queue_For_ThreatQ, '<QUEUES TO USE WITH EXTENSION (i.e.
Incidents, Incident Reports)>');
Set($ThreatQ_Host, '<YOUR THREATQ HOST/IP>');
Set($ThreatQ_Status, '<STATUS TO USE FOR NEW INDICATORS>');
Set($ThreatQ_Use_SSL, '<ENABLE TO DISABLE SSL WITH RT ('1' or '0')>');

Set($WebDomain, '<YOUR REQUEST TRACKER HOST/DOMAIN>');
```

17. Optional - Add logging to your Request Tracker instance, if not already done, by editing the same file referenced in step 16.

```
Set($LogToFile, 'debug');
Set($LogToFileNamed, 'rt.log');
Set($LogDir, '/opt/rt4/var/log');
```

18. If your Request Tracker instance uses HTTPS, run the following HTTPS patch command to enable it in the extension. If you do not, it may cause API access issues.

```
sed -i "s/http:\\\\\/\\\/https:\\\\\/\\\/" /opt/rt4/local/plugins/RT-Extension-
ThreatQ/bin/tq_request_tracker.py
```

19. Clear your mason cache:



```
rm -rf /opt/rt4/var/mason_data/obj
```

20. Restart your apache web server:

```
systemctl restart httpd.service
```

You will still need to [configure and then enable the connector and extension](#).


Configuration



ThreatQuotient does not issue API keys for third-party vendors. Contact the specific vendor to obtain API keys and other integration-related credentials.

To configure the integration:


1. Navigate to your integrations management page in ThreatQ.
2. Select the **Labs** option from the *Category* dropdown (optional).
3. Click on the integration entry to open its details page.
4. Enter the following parameters under the **Configuration** tab:

PARAMETER	DESCRIPTION
Host	The host of your Request Tracker instance.
Username	The username of the account you want to use to sync the data. <div>  This user must have API access privileges. </div>
Password	The password associated with the above username.
ThreatQ Host/IP	The host of your ThreatQ instance.

5. Review any additional settings, make any changes if needed, and click on **Save**.
6. Click on the toggle switch, located above the *Additional Information* section, to enable it.

Configuring the Extension

To configure this extension, you can either edit the `/opt/rt4/etc/RT_SiteConfig.pm` directly, or you can edit it through the Request Tracker web UI. To edit via the web UI, follow these steps.

1. Open to your Request Tracker home page (and login)
 2. In the navigation bar, go to `Admin -> Tools -> Edit ThreatQ Config`
 3. You will not be able to edit your settings from here:
 - You will not be able to edit passwords from this menu
 - The `ThreatQ_Use_SSL` option has 3 radio buttons. The first one is for `Yes` and the second one is for `No`. The third radio button should be ignored and not used.
 -  See [Step 16](#) in the `Installation` section to see how to fill out these fields.
 4. If you made any changes, click the `Save Changes` button
- You *do not* need to restart your Apache service.

Usage

Use the following command to execute the driver:

```
/opt/tqvenv/<environment_name>/bin/tqRequestTracker -v3 -ll /var/log/tq_labs/ -c /etc/tq_labs/
```

The connector should run on a CRON job. It is recommended to run the connector every 1 - 2 minutes to make sure syncing is as close to real time as possible. See the [CRON](#) section for more details

Request Tracker Extension

- This integration (extension) provides automatic and manual actions to sync Request Tracker tickets to ThreatQ. When you create a ticket in Request Tracker, it will automatically be added to ThreatQ, along with the data that is included in the ticket (if supported; see [Supported Custom Fields](#) section). The tickets will only be automatically synced if they are part of the Queues listed within the ThreatQ Config in Request Tracker.
- You can also manually sync tickets (in the configured Queues) from the ticket page. On the ticket page, find the actions menu. There you will see a ThreatQ menu item. Click it and select Sync Ticket. This will sync the ticket (and its' data) with ThreatQ. When it is done, you will be redirected to a page where you can see the progress log. From this page, you can return back to the ticket.
- Any time a ticket is updated, it will trigger the ticket syncing, and transfer your updates to ThreatQ.

Command Line Arguments

This connector supports the following custom command line arguments:

ARGUMENT	DESCRIPTION
-h, --help	Shows this help message and exits.
-ll LOGLOCATION, --loglocation LOGLOCATION	Sets the logging location for the connector. The location should exist and be writable by the current. A special value of 'stdout' means to log to the console (this happens by default).
-c CONFIG, --config CONFIG	This is the location of the configuration file for the connector. This location must be readable and writable by the current user. If no config file path is given, the current directory will be used. This file is also where some information from each run of the connector may be put (last run time, private oauth, etc.)

ARGUMENT	DESCRIPTION
<code>-v {1,2,3}, --verbosity {1,2,3}</code>	This is the logging verbosity level where 3 means everything. The default setting is 1 (Warning).
<code>-n, --name</code>	This allows you to change the name of the connector.
<code>-d, --no-differential</code>	If exports are used in this connector, this will turn 'off' the differential flag for the execution. This allows debugging and testing to be done on export endpoints without having to rebuild the exports after the test. THIS SHOULD NEVER BE USED IN PRODUCTION.
<code>-ep, --external-proxy</code>	This enables a proxy to be used to contact the internet for the data required by this connector. This specifies an internet facing proxy, NOT a proxy to the TQ instance.
<code>-ds, --disable-ssl</code>	Disable SSL verification

CRON

Automatic CRON configuration has been removed from this script. To run this script on a recurring basis, use CRON or some other jobs scheduler. The argument in the CRON script must specify the config and log locations.

Add an entry to your Linux crontab to execute the connector at a recurring interval. Depending on how quickly you need updates, this can be run multiple times a day (no more than once an hour) or a few times a week.

In the example below, the command will execute the connector every two hours.

1. Log into your ThreatQ host via a CLI terminal session.
2. Enter the following command:

```
crontab -e
```

This will enable the editing of the crontab, using vi. Depending on how often you wish the cronjob to run, you will need to adjust the time to suit the environment.

3. Enter the commands below:

Every Minute Example

```
*/1* * * * /opt/tqenv/<environment_name>/bin/tqRequestTracker -c /  
etc/tq_labs/ -ll /var/log/tq_labs/ -v3
```

4. Save and exit CRON.

Supported Custom Fields

The following is a list of support custom fields:

- Description
- Resolution
- Classification
- IP (multiple values)
Not including IP ranges or CIDR Blocks.
- Domains (multiple values)
- FileNames (multiple values)
- HASH (multiple values)
- Sender (multiple value)
- Systems Re-Imaged (multiple values)
- External Ticket Number (single value)
- Tags (multiple values)
- Threat Campaign (single value)
- URL (multiple value)
- Threat Actors (multiple values)
- Threat TTP (multiple values)
- ThreatQ Link (single value)
- ThreatQ Attributes (multiple values)



Shows TQ attributes in Request Tracker.

- ThreatQ Indicators (multiple values)



Shows TQ indicators, their type, and a link to TQ in Request Tracker.

Enabling Custom Fields

Enabling custom fields will require an administrative account (after installation of the ThreatQ extension), but it will allow the usage of the ThreatQ custom fields within tickets. Upon installation of the extension/plugin, there will be 3 custom fields added: ThreatQ Attributes, ThreatQ Link, and ThreatQ Indicators. These custom fields are optional to add to your queues, but are recommended. Here is how to do that.

1. Open to your Request Tracker home page.
2. In the navigation bar, go to Admin -> Custom Fields -> Select.



This will bring you to a page where you see a list of all custom fields. You should see the ThreatQ custom fields in this list, but they will not be assigned to any queues.

3. For each ThreatQ custom field, click on the name.

-
4. In the secondary navigation menu (on the right), select **Applies to**.

This will bring you to a page where you can "apply" the custom field to queues.

5. Select the queues that you want to add the custom field to, and click **Submit**.
6. Repeat for each custom field from ThreatQ.

Upgrading the Extension

If there is a new version of this extension, follow these steps to upgrade your ThreatQ Extension for Request Tracker

1. SSH into your Request Tracker instance and then SCP over the new version of the ThreatQ Extension
2. Extract the extension and then navigate into its' directory
3. Make yourself root

```
sudo su
```

4. Install the extension into Request Tracker

```
perl Makefile.PL $> make $> make install
```

5. Clear your mason cache

```
rm -rf /opt/rt4/var/mason_data/obj
```

6. Restart your Apache service

```
systemctl restart httpd.service
```

Change Log

Request Tracker Connector

- **Version 2.1.0**
 - Added Python 3 functionality.
 - Added functionality to delete any object from TQ and it will be reflected in RT.
 - Fixed sync for indicators, adversaries, attributes, and events.
- **Version 2.0.0**
 - Initial Release

Request Tracker Extension

- **Version 1.1.4**
 - Fixed delete of indicators in RT; can now delete indicators from RT and thereby from TQ's ticket event.
 - Added get_utc method to tq_request_tracker.
 - Fixed sync of indicators, attributes, actors/adversaries, campaigns/events.
- **Version 1.0.0**
 - Initial Release