ThreatQuotient

Active Directory Connector Guide

Version 1.1.0

July 19, 2021

ThreatQuotient
11400 Commerce Park Dr., Suite 200
Reston, VA 20191

Support
Email: support@threatq.com
Web: support.threatq.com
Phone: 703.574.9893

Contents

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.

Versioning

- Current integration version: `1.1.0`
- Supported on ThreatQ versions >= `4.34.0`

There are two versions of this integration:

- Python 2 version
- Python 3 version

Introduction

The Active Directory for ThreatQuotient Connector provides a way for users to import their active directory identities into ThreatQ.

Installation

The connector can be installed from the ThreatQuotient repository with YUM credentials or offline via a .whl file.

> **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. Install the connector using one of the following methods:

   ThreatQ Repository

   a. Run the following command:

   ```
   <> pip install tq_conn_active_directory
   ```

   Offline via .whl file

   To install this connector from a wheel file, the wheel file (.whl) will need to be copied via SCP into your ThreatQ instance.

   a. Download the connector whl file with its dependencies:

   ```
   <> mkdir /tmp/tq_conn_active_directory

      pip download tq_conn_active_directory -d

      /tmp/tq_conn_active_directory/
   ```

   b. Archive the folder with the .whl files:

   ```
   <> tar -czvf tq_conn_active_directory.tgz /tmp/
      tq_conn_active_directory/
   ```

   c. Transfer all the whl files, the connector and all the dependencies, to the ThreatQ instance.

   d. Open the archive on ThreatQ:

   ```
   <> tar -xvf tq_conn_active_directory.tgz
   ```

e. Install the connector on the ThreatQ instance.

> The example assumes that all the whl files are copied to `/tmp/conn` on the ThreatQ instance.

```
<> pip install /tmp/conn/ tq_conn_active_directory-<version>-
   <python version>-none-any.whl --no-index --find-links /
   tmp/conn/
```

> A driver called `tq-conn-active-directory` will be installed. After installing with `pip` or `setup.py`, a script stub will appear in `/usr/bin/tq-conn-active-directory`.

2. 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
```

3. Perform an initial run using the following command:

```
<> tq-conn-active-directory -v3 -ll /var/log/tq_labs/ -c /etc/
   tq_labs/
```

4. 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.
Client ID	This is the OAuth id that can be found at Settings Gear → User Management → API details within the user's details.
Email Address	This is the User in the ThreatQ System for integrations.
Password	The password for the above ThreatQ account.

PARAMETER	DESCRIPTION
Status	This is the default status for objects that are created by this Integration.

Example Output

```
tq-conn-active-directory -v3 -ll /var/log/tq_labs/ -c /etc/tq_labs/
ThreatQ Host: <ThreatQ Host IP or Hostname>
Client ID: <ClientID>
E-Mail Address: <EMAIL ADDRESS>
Password: <PASSWORD>
Status: Review
Connector configured. Set information in UI
```

You will still need to configure and then enable the connector.

Configuration

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

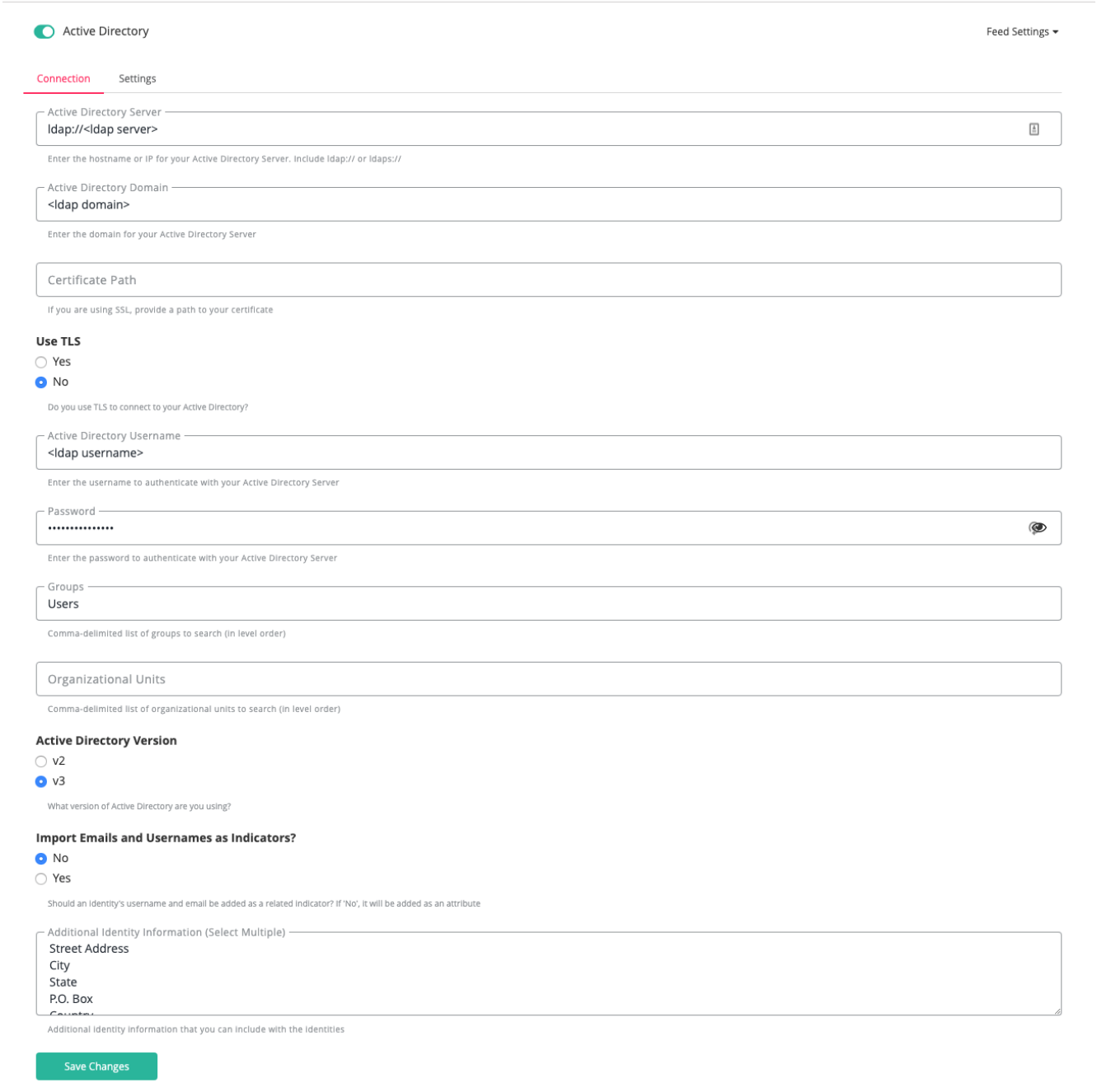
To configure the integration:

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

PARAMETER	DESCRIPTION
Active Directory Server	The address to your LDAP server. ldap:// or ldaps:// must be included.
Active Directory Domain	The domain used by your LDAP server
Certificate Path	The path to your certificate associated with your LDAP server. See the Generating a Certificate section for more details. This path must be accessible by the connector. This path is the full path to the .pem file. Leaving this blank will result in not using a certificate.
Use TLS	Enable/disable TLS for your connection.

PARAMETER	DESCRIPTION
Active Directory Username	The username to use to authenticate with your LDAP server.
Active Directory Password	The password used to authenticate with your LDAP server.
Common Names	A list of common names to pull users from. The connector pulls all subtrees of a common name, configure accordingly. This list must be in level order (top-most to bottom-most)
Organizational Units	A list of organizational units to pull users from. The connector pulls all subtrees of a OU, configure accordingly. This list must be in level order (top-most to bottom-most)
Active Directory Version	Set your version of Active Directory. The default setting is **v3**.
Import Emails and Usernames as Indicators	Configure where email addresses and username are imported as indicators into ThreatQ. Emails and usernames are added as attributes to identify objects by default. Selecting **Yes** will result in emails and usernames being imported as related indicaors.
Additional Identity Information	Enter any additional indentity infomration to import in this field. By default, basic identity information such as username,

PARAMETER	DESCRIPTION
	name, manager, position, company, office location, etc. is included.



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

Usage

Use the following command to execute the driver:

```
<> tq-conn-active-directory -v3 -ll /var/log/tq_labs/ -c /etc/
   tq_labs/
```

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.)
`-v {1,2,3}`, `--verbosity {1,2,3}`	This is the logging verbosity level where **3** means everything. The default setting is **1** (Warning).
`-n, --name`	This allows you to change the name of the connector.
`-d, --no-differential`	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

ARGUMENT	DESCRIPTION
	exports after the test. THIS SHOULD NEVER BE USED IN PRODUCTION.
`-ep, --external-proxy`	This enables a proxy to be used to connect to the internet for the data required by this connector. This specifies an internet facing proxy, NOT a proxy to the TQ instance.
`-cache CACHE, --cache CACHE, -cc, --clear-cache`	

Generating a Certificate

If you are using SSL/TLS with your LDAP server (ldaps://), you can enable the use of a certificate by generating one, then entering the full path into the configuration. Use the following command to generate a certificate:

```
<> openssl s_client -connect <LDAP SERVER>:636 -showcerts -tls1 < /
   dev/null > cacerts.pem 2> /dev/null
```

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 2 Hours Example

   ```
   <> 0 */2 * * * tq-conn-active-directory -c /etc/tq_labs/ -ll /
      var/log/tq_labs/ -v3
   ```

4. Save and exit CRON.

Change Log

- **Version 1.1.0**
  - Added Python 3 support.
- **Version 1.0.0**
  - Initial Release