Connect to Reveal(x) 360 from self-managed sensors through the REST API

Published: 2021-09-04

The ExtraHop REST API enables you to automate connections for a large number of self-managed sensors to Reveal(x) 360 with a script. Self-managed sensors include on-premises Discover appliances or instances deployed on cloud service providers such as AWS, Azure, and Google Cloud Platform (GCP).

This guide provides instructions for the REST API Explorer, so you can test the REST operation, and an example Python script that you can modify with your environment variables.

Note: You cannot connect Trace appliances through the REST API. For information about connecting Trace appliances, see Connect to Reveal(x) 360 from self-managed sensors.

Before you begin

- Familiarize yourself with the ExtraHop REST API Guide to learn how to navigate the ExtraHop REST API Explorer.
- You must have an Okta user account with OktaAdmin privileges to configure Reveal(x) 360. Details for setting up this account are in the introduction email sent from ExtraHop Networks.
- You must generate tokens through Reveal(x) 360 for each sensor that you want to connect. For more information, see Connect to Reveal(x) 360 from self-managed sensors.
- You must log in to the ExtraHop system with an account that has unlimited privileges to generate an API key.
- You must have a valid API key to make changes through the REST API and complete the procedures below. (See Generate an API key.)

Connect to Reveal(x) 360 through the REST API Explorer

1. In a browser, navigate to the REST API Explorer.

   The URL is the hostname or IP address of your ExtraHop system, followed by `/api/v1/explore/`. For example, if your hostname is seattle-eda, the URL is `https://seattle-eda/api/v1/explore/`.
2. Click **Enter API Key** and then paste or type your API key into the **API Key** field.
3. Click **Authorize** and then click **Close**.
4. Click **Cloud** and then click **POST /cloud/connect**.
5. Click **Try it out**.
6. In the body field, replace `string` with the token you generated from Reveal(x) 360, as shown in the following example:

   ```
   {
       "cloud_token": "561b85-e9092a3a-343fcb03-78c72777-8db70bbd"
   }
   ```

   The Server response section displays a 201 status code.

Python script example

The following Python script connects your sensor to Reveal(x) 360 by reading tokens and API keys from a CSV file that meets the following specifications:

- The CSV file must not contain a header row.
- Each row of the CSV file must contain the following three columns in the specified order:

The sensor URL, including the `https://` schema specification	The sensor API key	The token you generated from Reveal(x) 360

- The CSV file must be named `sensors.csv` and stored in the same directory as the script.

```
#!/usr/bin/python3

import requests
import csv
import json

SENSORS_LIST = 'sensors.csv'

# Read sensor hostnames and connection tokens from CSV
sensors = []
with open(SENSORS_LIST, 'r') as f:
    reader = csv.reader(f)
    for row in reader:
        sensor = {
            'host': row[0],
            'api_key': row[1],
            'token': row[2]
        }
        sensors.append(sensor)

# Function that connects a sensor to CCP
def connectSensor(sensor):
    url = sensor['host'] + '/api/v1/cloud/connect'
    headers = {'Authorization': 'ExtraHop apikey=%s' % sensor['api_key']}
    data = {
        "cloud_token": sensor['token']
    }
    r = requests.post(url, headers=headers, data=json.dumps(data))
    if r.status_code == 201:
        print('Successfully paired ' + sensor['host'])
    else:
        print('Error! Failed to pair ' + sensor['host'])
        print(r.text)

for sensor in sensors:
    connectSensor(sensor)
```

Note: If the script returns an error message that the SSL certificate verification failed, make sure that a trusted certificate has been added to your ExtraHop system. Alternatively, you can add the `verify=False` option to bypass certificate verification. However, this method is not secure and is not recommended. The following code sends an HTTP GET request without certificate verification:

```
requests.get(url, headers=headers, verify=False)
```