Create a device group through the REST API

Published: 2024-03-26

You can create a large number of complex device groups through the REST API by referencing a CSV file exported from a third-party application. In this topic, we show methods for creating a device group through both the ExtraHop REST API Explorer and a Python script.

Before you begin

- For sensors and ECA VMs, you must have a valid API key to make changes through the REST API and complete the procedures below. (See Generate an API key.)
- For Reveal(x) 360, you must have valid REST API credentials to make changes through the REST API and complete the procedures below. (See Create REST API credentials.)

Create a device group 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 sensor or console, 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 **Device Group** and then click **POST /devicegroups**.
5. Click **Try it out**.

   The JSON schema is automatically added to the body parameter text box.
6. In the body field, specify properties for the device group that you want to create.

   For example, the following body creates a device group that includes CIDR blocks `192.168.0.0/26`, `192.168.0.64/27`, and `192.168.0.96/30`:

```
{
    "name": "New group",
    "description": "A newly created group",
    "filter": {
        "rules": [
            {
                "field": "ipaddr",
                "operand": "192.168.0.0/26",
                "operator": "="
            },
            {
                "field": "ipaddr",
                "operand": "192.168.0.64/27",
                "operator": "="
            },
            {
                "field": "ipaddr",
                "operand": "192.168.0.96/30",
                "operator": "="
            }
        ],
        "operator": "or"
    }
}
```

7. Click **Send Request**.

Retrieve and run the example Python script

The ExtraHop GitHub repository contains an example Python script that creates device groups by reading criteria from a CSV file that meets the following specifications:

1. Go to the ExtraHop code-examples GitHub repository and download the `create_device_groups/create_device_groups.py` file to your local machine.
2. In the directory you copied the `create_device_groups.py` to, create 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:

Device group name	Description	IP address or CIDR block

   - Each column after the first required three columns must specify an IP address or CIDR block for the device group.

     Note: You cannot specify more than 1000 IP addresses or CIDR blocks for a device group.

   Note: For an example of a compatible CSV file, see the create_device_groups/device_group_list.csv file in the ExtraHop code-examples GitHub repository.
3. In a text editor, open the `create_device_groups.py` file and replace the configuration variables with information from your environment.
   - For sensors and ECA VMs, specify the following configuration variables:
     - **HOST:** The IP address or hostname of the sensor or ECA VM.
     - **API_KEY:** The API key.
     - **CSV_FILE:** The file that contains the list of device groups.
   - For Reveal(x) 360, specify the following configuration variables:
     - **HOST:** The hostname of the Reveal(x) 360 API. This hostname is displayed in the Reveal(x) 360 API Access page under API Endpoint. The hostname does not include the `/oauth2/token`.
     - **ID:** The ID of the Reveal(x) 360 REST API credentials.
     - **SECRET:** The secret of the Reveal(x) 360 REST API credentials.
     - **CSV_FILE:** The file that contains the list of device groups.
4. Run the following command:

```
python create_device_groups.py
```

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 sensor or console. Alternatively, you can add the `verify=False` option to bypass certificate verification. However, this method is not secure and not recommended. The following code sends an HTTP GET request without certificate verification:

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