Specify custom device makes and models through the REST API

Published: 2025-03-25

The ExtraHop REST API enables you to specify custom makes and models for devices. You can update devices through the REST API Explorer or automate the procedure by reading device criteria from a CSV or similar file through a REST API script.

Before you begin

- For sensors and the ExtraHop console, 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 RevealX 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.)

Specify a custom make and model through the REST API Explorer

Retrieve the ID of the device

Before you can specify a custom make and model for a device, you must retrieve the REST API ID of the device.

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. Enter your REST API credentials.
   - For sensors and the ExtraHop console, click **Enter API Key** and then paste or type your API key into the **API Key** field.
   - For RevealX 360, click **Enter API Credentials** and then paste or type the ID and secret of your API credentials into the **ID** and **Secret** fields.
3. Click **Authorize** and then click **Close**.
4. Click **POST /devices/search**.
5. Click **Try it out**.

   The JSON schema is automatically added to the body parameter text box.
6. In the body text box, type search criteria that selects the device.
   The following search criteria returns a device with an IP address of 10.10.10.200:

   ```
   {
     "filter": {
       "field": "ipaddr",
       "operand": "10.10.10.200",
       "operator": "="
     }
   }
   ```

   For more information about device search filters, see Operand values for device search.
7. Click **Send Request**.

   In the Response body section, note the `id` field of the device.

Specify a high value device

1. Click **PATCH /devices/{id}**.
2. Click **Try it out**.
3. In the **body** field, type the following JSON object:

```
{
    "custom_criticality": "critical"
}
```

4. In the **id** field, type the ID of the device that you retrieved in the previous procedure.
5. Click **Send Request**.

   If the request is successful, a 204 response code appears in the Server response section.

Retrieve and run the example Python script

The ExtraHop GitHub repository contains an example Python script that reads custom makes and models from a CSV file and adds them to devices with specified IP addresses.

1. Go to the ExtraHop code-examples GitHub repository and download the contents of the `specify_custom_make_model` directory to your local machine.
2. In a text editor, open the `custom_config.csv` file and add the IP address for each device you want to update along with the custom makes and models for each device.

   Note: The file contains two example entries. The script ignores the header row.
3. In a text editor, open the `specify_custom_make_model.py` file and replace the configuration variables with information from your environment.
   - For sensors and ExtraHop consoles, specify the following configuration variables:
     - **HOST:** The IP address or hostname of the sensor or ExtraHop console.
     - **API_KEY:** The API key.
   - For RevealX 360, specify the following configuration variables:
     - **HOST:** The hostname of the RevealX 360 API. This hostname is displayed in the RevealX 360 API Access page under API Endpoint. The hostname does not include the `/oauth2/token`.
     - **ID:** The ID of the RevealX 360 REST API credentials.
     - **SECRET:** The secret of the RevealX 360 REST API credentials.
4. Run the following command:

```
python3 specify_custom_make_model.py
```

   Note: If the script returns an error message that the TLS 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 is not recommended. The following code sends an HTTP GET request without certificate verification:

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