

# Create custom devices through the REST API

Published: 2020-02-22

You can create custom devices through the REST API that track network traffic across multiple IP addresses and ports. For example, you might want to add a custom device for each branch office in ExtraHop system. If you create the devices through a script, you can read the list of devices from a CSV file. In this topic, we will demonstrate methods for both the REST API and the ExtraHop REST API Explorer.

## Before you begin

- You must log into the ExtraHop appliance 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](#).)
- Familiarize yourself with the [ExtraHop REST API Guide](#) to learn how to navigate the ExtraHop REST API Explorer.

## Create a custom device

Before you can associate a custom device with an IP address, you must create the device in the ExtraHop system.

1. In the REST API Explorer, click **Custom Device**, and then click **POST /customdevices**.
2. In the body field, specify properties for the custom device that you want to create.

```
{
  "description": "The location of our office in Washington",
  "name": "Seattle"
}
```

## Add custom device criteria

When you initially add a custom device, it is an empty object in the ExtraHop system. To associate the custom device with a real-world device or application, you must add one or more IP addresses.

1. In the REST API Explorer, click **Custom Device**, and then click **POST /customdevices/{id}/criteria**.
2. In the body field, specify criteria for the custom device that you created.

For example, the following body matches the custom device to the CIDR block "192.168.0.0/24":

```
{
  "ipaddr": "192.168.0.0/24"
}
```

3. In the ID field, specify the numeric ID for the custom device.




**Tip:** You can view the IDs of custom devices by navigating to **Custom Device > GET/customdevices** and clicking **Try it out!**. The ID appears in the response body under **id**.

## Python script example

This example python script creates custom devices by reading criteria from a CSV file. Each row of the CSV file must contain the following columns in the specified order:

Name	ID	Description	IP address or CIDR block
------	----	-------------	--------------------------

 **Note:** The script does not accept a header row in the CSV file. There is no limit to the number of columns in the table; each column after the first four specifies an additional IP address for the device. The first four columns are required for each row.

For example, the following CSV list contains criteria for offices in France, Holland, and California:

```
France,francehq,The location of our office in
France,192.168.0.103,192.168.0.105,192.168.0.101
Holland,hollandhq,The location of our office in Holland,192.168.0.102
California,californiahq,The location of our office in
California,192.168.0.104,192.168.0.103
```

The script includes the following configuration variables:

- **HOST:** The IP address or hostname of the Discover appliance
- **APIKEY:** The API key
- **CSV\_FILE:** The path of the CSV file relative to the location of the script file

```
#!/usr/bin/env python2

import json
import httplib
import ssl
import csv
import os.path

HOST = 'example.extrahop.com'
APIKEY = 'f6876657888a7c1f24ac77827'
CSV_FILE = 'device_list.csv'

context = ssl.SSLContext(ssl.PROTOCOL_TLSv1_2)
headers = {'Content-Type': 'application/json',
          'Accept': 'application/json',
          'Authorization': 'ExtraHop apikey=%s' % APIKEY}

def readCSV():
    devices = []
    with open(CSV_FILE, 'rb') as f:
        reader = csv.reader(f)
        for row in reader:
            device = {}
            device['name'] = row.pop(0)
            device['extrahop_id'] = row.pop(0)
            device['description'] = row.pop(0)
            device['ipaddr'] = row
            devices.append(device)
    return devices

def createDevice(device):
    ips = device.pop('ipaddr')
    conn = httplib.HTTPSConnection(HOST, context=context)
    conn.request('POST', '/api/v1/customdevices', body=json.dumps(device),
                headers=headers)
    resp = conn.getresponse()
    if resp.status != 201:
        print "Could not create device: " + device['name']
        print "      " + json.loads(resp.read())['error_message']
        return -1, ips
    else:
```

```
    print "Created custom device: " + device['name']
    device_id = os.path.basename(resp.getheader('location'))
    return device_id, ips

def addIPs(device_id, ips, name):
    url = '/api/v1/customdevices/' + device_id + '/criteria'
    for ip in ips:
        conn = httplib.HTTPSConnection(HOST, context=context)
        body = {"ipaddr": ip}
        conn.request('POST', url, body=json.dumps(body), headers=headers)
        resp = conn.getresponse()
        if resp.status != 201:
            print "    Could not add IP " + ip + " for " + name
            print "    " + json.loads(resp.read())['error_message']
        else:
            print "    Added IP " + ip + " for " + name

devices = readCSV()
for device in devices:
    device_id, ips = createDevice(device)
    if device_id == -1:
        continue
    addIPs(device_id, ips, device['name'])
```