ExtraHop 8.9
ExtraHop REST API Guide

For more documentation, see https://docs.extrahop.com/.

Published: 2022-11-08

ExtraHop Networks
Seattle, WA 98101
877-333-9872 (US)
+44 (0)203 7016850 (EMEA)
+65-31585513 (APAC)
www.extrahop.com

Contents

Introduction to the ExtraHop REST API

The ExtraHop REST API enables you to automate administration and configuration tasks on your ExtraHop system. You can send requests to the ExtraHop API through a Representational State Transfer (REST) interface, which is accessed through resource URIs and standard HTTP methods.

When a REST API request is sent over HTTPS to an ExtraHop system, that request is authenticated and then authorized through an API key. After authentication, the request is submitted to the ExtraHop system and the operation completes.

Each ExtraHop system provides access to the built-in ExtraHop REST API Explorer, which enables you to view all of the available system resources, methods, properties, and parameters. The REST API Explorer also enables you to send API calls directly to your ExtraHop system.

Note: This guide is intended for an audience that has a basic familiarity with software development and the ExtraHop system.

ExtraHop API requirements

Before you can begin writing scripts for the ExtraHop REST API or performing operations through the REST API Explorer, you must meet the following requirements:

- Your ExtraHop system must be configured to allow API key generation for the type of user you are (remote or local).
- You must generate a valid API key.
- You must have a user account on the ExtraHop system with appropriate privileges set for the type of tasks you want to perform.

Access and authenticate to the ExtraHop REST API

Administrators, or users with unlimited privileges, control whether users can generate API keys. For example, you can prevent remote users from generating keys or you can disable API key generation entirely. When this functionality is enabled, API keys are generated by users and can be viewed only by the user who generated the key.

Note: Administrators set up user accounts and assign privileges, but then users generate their own API keys. Users can delete API keys for their own account, and users with unlimited privileges can delete API keys for any user. For more information, see Users and user groups.

After you generate an API key, you must append the key to your request headers. The following example shows a request that retrieves metadata about the firmware running on the ExtraHop system:

```
curl -i -X GET --header "Accept: application/json" \
--header "Authorization: ExtraHop apikey=2bc07e55971d4c9a88d0bb4d29ecbb29" \
"https://<hostname-or-IP-of-your-ExtraHop-system>/api/v1/extrahop"
```

Privilege levels

User privilege levels determine which ExtraHop system and administration tasks the user can perform through the ExtraHop REST API.

You can view the privilege levels for users through the `granted_roles` and `effective_roles` properties. The `granted_roles` property shows you which privilege levels are explicitly granted to the user. The `effective_roles` property shows you all privilege levels for a user, including those received outside of the granted role, such as through a user group.

The `granted_roles` and `effective_roles` properties are returned by the following operations:

- GET /users
- GET /users/{username}

The `granted_roles` and `effective_roles` properties support the following privilege levels. Note that the type of tasks for each ExtraHop system vary by the available resources listed in the REST API Explorer.

Privilege level	Actions allowed
"system": "full"	• Enable or disable API key generation for the ExtraHop system. • Generate an API key. • View the last four digits and description for any API key on the system. • Delete API keys for any user. • View and edit cross-origin resource sharing. • Transfer ownership of any non-system dashboard to another user. • Perform any administration task available through the REST API. • Perform any ExtraHop system task available through the REST API.
"write": "full"	• Generate your own API key. • View or delete your own API key. • Change your own password, but you cannot perform any other administration tasks through the REST API.

Privilege level	Actions allowed
	• Perform any ExtraHop system task available through the REST API.
"write": "limited"	• Generate an API key. • View or delete their own API key. • Change your own password, but you cannot perform any other administration tasks through the REST API. • Perform all GET operations through the REST API. • Modify the sharing status of dashboards that you are allowed to edit. • Delete dashboards and activity maps that you own. • Perform metric and record queries.
"write": "personal"	• Generate an API key. • View or delete your own API key. • Change your own password, but you cannot perform any other administration tasks through the REST API. • Perform all GET operations through the REST API. • Delete dashboards and activity maps that you own. • Perform metric and record queries.
"metrics": "full"	• Generate an API key. • View or delete your own API key. • Change your own password, but you cannot perform any other administration tasks through the REST API. • View dashboards and activity maps shared with you. • Perform metric and record queries.
"metrics": "restricted"	• Generate an API key. • View or delete your own API key. • Change your own password, but you cannot perform any other administration tasks through the REST API. • View dashboards and activity maps shared with you.
"packets": "full"	• View and download packets from an ExtraHop system through the `GET/packetcaptures/{id}` operation. This is an add-on privilege that can be granted to a user with one of the following privilege levels: • "write": "full" • "write": "limited" • "write": "personal" • "write": null • "metrics": "full" • "metrics": "restricted"
"packets": "full_with_keys"	• View and download packets from an ExtraHop system through the `GET/packetcaptures/{id}` operation. This is an add-on privilege that can be granted to a user with one of the following privilege levels:

Privilege level	Actions allowed
	• "write": "full" • "write": "limited" • "write": "personal" • "write": null • "metrics": "full" • "metrics": "restricted"
"detections": "full"	• View detections in the ExtraHop system. This is an add-on privilege that can be granted to a user with one of the following privilege levels: • "write": "full" • "write": "limited" • "write": "personal" • "write": null • "metrics": "full" • "metrics": "restricted"
"detections": "none"	• No access to detections. This is an add-on privilege that can be granted to a user with one of the following privilege levels: • "write": "full" • "write": "limited" • "write": "personal" • "write": null • "metrics": "full" • "metrics": "restricted"

Manage API key access

Users with unlimited privileges can configure whether users can generate API keys for the ExtraHop system. You can allow only local users to generate keys, or you can also disable API key generation entirely.

Users must generate an API key before they can perform operations through the ExtraHop REST API. Keys can be viewed only by the user who generated the key or system administrators with unlimited privileges. After a user generates an API key, they must append the key to their request headers.

1. Log in to the Administration settings on the ExtraHop system through `https://<extrahop-hostname-or-IP-address>/admin`.
2. In the Access Settings section, click **API Access**.
3. In the Manage API Access section, select one of the following options:
   - **Allow all users to generate an API key**: Local and remote users can generate API keys.
   - **Only local users can generate an API key**: Remote users cannot generate API keys.
   - **No users can generate an API key**: No API keys can be generated by any user.
4. Click **Save Settings**.

Generate an API key

You must generate an API key before you can perform operations through the ExtraHop REST API. Keys can be viewed only by the user who generated the key or by system administrators with unlimited privileges. After you generate an API key, add the key to your request headers or the ExtraHop REST API Explorer.

Before you begin
Make sure the ExtraHop system is configured to allow API key generation.

1. In the Access Settings section, click **API Access**.
2. In the Generate an API Key section, type a description for the new key, and then click **Generate**.
3. Scroll down to the API Keys section, and copy the API key that matches your description.

You can paste the key into the REST API Explorer or append the key to a request header.

Configure cross-origin resource sharing (CORS)

Cross-origin resource sharing (CORS) allows you to access the ExtraHop REST API across domain-boundaries and from specified web pages without requiring the request to travel through a proxy server.

You can configure one or more allowed origins or you can allow access to the ExtraHop REST API from any origin. Only administrative users with unlimited privileges can view and edit CORS settings.

1. In the **Access Settings** section, click **API Access**.
2. In the CORS Settings section, specify one of the following access configurations.
   - To add a specific URL, type an origin URL in the text box, and then click the plus (+) icon or press ENTER.

     The URL must include a scheme, such as `HTTP` or `HTTPS`, and the exact domain name. You cannot append a path; however, you can provide a port number.
   - To allow access from any URL, select the Allow API requests from any Origin checkbox.

     Note: Allowing REST API access from any origin is less secure than providing a list of explicit origins.
3. Click **Save Settings** and then click **Done**.

Set up an SSL certificate

Before making requests to an ExtraHop system with a self-signed certificate, you must set up an SSL certificate for each user who will access the ExtraHop system from a particular computer.

In each of the following examples, replace {HOST} with the hostname of your ExtraHop system.

Note: The SSL certificate applies only to the user performing the command. Each user must run the command with their login credentials to set up the SSL certificate.

Set up SSL through Windows PowerShell

```
Invoke-WebRequest "http://{HOST}/public.cer" -OutFile ($env:USERPROFILE +
"\ex.cer"); Import-Certificate ($env:USERPROFILE + "\ex.cer")
-CertStoreLocation Cert:\CurrentUser\Root
```

Set up SSL through OS X

```
curl -O http://{HOST}/public.cer; security add-trusted-cert -r trustRoot -k
```

```
~/Library/Keychains/login.keychain public.cer
```

Learn about the REST API Explorer

The REST API Explorer is a web-based tool that enables you to view detailed information about the ExtraHop REST API resources, methods, parameters, properties, and error codes. Code samples are available in Python, cURL, and Ruby for each resource. You also can perform operations directly through the tool.

Open the REST API Explorer

You can open the REST API Explorer from the Administration settings or through the following URL:

```
https://<extrahop-hostname-or-ip-address>/api/v1/explore/
```

1. Log in to the Administration settings on the ExtraHop system through `https://<extrahop-hostname-or-IP-address>/admin`.
2. From the Access Setting section, click **API Access**.
3. On the API Access page, click **REST API Explorer**.

   The REST API Explorer opens in your browser.

View operation information

From the REST API Explorer, you can click any operation to view configuration information for the resource.

The following table provides information about the sections available for resources in the REST API Explorer. Section availability varies by HTTP method. Not all methods have all of the sections listed in the table.

Section	Description
Body Parameters	Provides all of the fields for the request body and supported values for each field.
Parameters	Provides information about the available query parameters.
Responses	Provides information about the possible HTTP status codes for the resource. If you click **Send Request**, this section also includes the response from the server and the cURL, Python, and Ruby syntax required to send the specified request.

Identify objects on the ExtraHop system

Objects on the ExtraHop system can be identified by any unique value, such as the IP address, MAC address, name, or system ID. However, to perform API operations on a specific object, you must locate the object ID. You can easily locate the object ID through the following methods in the REST API Explorer.

- The object ID is provided in the headers returned from a POST request. For example, if you send a POST request to create a page, the response headers display a location URL.

The following request returned the location for the newly created tag as `/api/v1/tags/1` and the ID for the tag as 1.

```
{
 "date": "Tue, 09 Nov 2021 18:21:00 GMT ",
 "via": "1.1 localhost",
 "server": "Apache",
 "content-type": "text/plain; charset=utf-8",
 "location": "/api/v1/tags/1",
 "cache-control": "private, max-age=0",
 "connection": "Keep-Alive",
 "keep-alive": "timeout=90, max=100",
 "content-length": "0"
}
```

- The object ID is provided for all objects returned from a GET request. For example, if you perform a GET request on all devices, the response body contains information for each device, including the ID.

  The following response body displays an entry for a single device, with an ID of 10212:

```
{
 "mod_time": 1448474346504,
 "node_id": null,
 "id": 10212,
 "extrahop_id": "test0001",
 "description": null,
 "user_mod_time": 1448474253809,
 "discover_time": 1448474250000,
 "vlanid": 0,
 "parent_id": 9352,
 "macaddr": "00:05:G3:FF:FC:28",
 "vendor": "Cisco",
 "is_l3": true,
 "ipaddr4": "10.10.10.5",
 "ipaddr6": null,
 "device_class": "node",
 "default_name": "Cisco5",
 "custom_name": null,
 "cdp_name": "",
 "dhcp_name": "",
 "netbios_name": "",
 "dns_name": "",
 "custom_type": "",
 "analysis_level": 1
},
```

- The object ID is provided in the URL for most objects. For example, in the ExtraHop system, click on **Assets**, and then **Devices**. Select any device and view the URL. In the following example, the URL for the device page shows Oid=10180.

```
https://10.10.10.205/extrahop/#/Devices?details=true&device
Oid=10180&from=6&interval_type=HR&until=0&view=l2stats
```

  To perform specific requests for that device, add 10180 to the id field in the REST API Explorer or to the body parameter in your request.

  The URL for dashboards displays a short_code, which appears after /Dashboard. When you add the short_code to the REST API Explorer or to your request, you must prepend a tilde to the short code.

In the following example, kmC9Y is the short_code. To perform requests for this dashboard, add ~kmC9Y as the value for the short_code field.

```
https://10.10.10.205/extrahop/#/Dashboard/kmC9Y/?from=6&interval_
type=HR&until=0
```

You can also find the short_code and dashboard ID in the Dashboard Properties for any dashboard, which can be accessed from the command menu ⋮. Some API operations, such as DELETE, require the dashboard ID.

ExtraHop API resources

You can perform operations on the following resources through the ExtraHop REST API. You also can view more detailed information about these resources, such as available HTTP methods, query parameters, and object properties in the REST API Explorer.

Activity Map

An activity map is a dynamic visual representation of the L4-L7 protocol activity between devices in your network. Create a 2D or 3D layout of device connections in real-time to learn about the traffic flow and relationships between devices.

Here are some important considerations about activity maps:

- You can only create activity maps for devices in Standard Analysis and Advanced Analysis. Discovery Mode devices are not included in activity maps. For more information, see Analysis levels.
- If you create an activity map for a device, activity group, or device group with no protocol activity in the selected time interval, the map appears without any data. Change the time interval or your origin selection and try again.
- You can create an activity map in a console to view device connections across all of your sensors.

To learn about configuring and navigating activity maps, see Activity maps.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /activitymaps	Retrieve all activity maps.
POST /activitymaps	Create a new activity map.
POST /activitymaps/query	Perform a network topology query, which returns activity map data in flat file content.
DELETE /activitymaps/{id}	Delete a specific activity map.
GET /activitymaps/{id}	Retrieve a specific activity map.
PATCH /activitymaps/{id}	Update a specific activity map.
POST /activitymaps/{id}/query	Perform a topology query for a specific activity map, which returns activity map data in flat file content.
GET /activitymaps/{id}/sharing	Retrieve the users and their sharing permissions for a specific activity map.
PATCH /activitymaps/{id}/sharing	Update the users and their sharing permissions for a specific activity map.
PUT /activitymaps/{id}/sharing	Replace the users and their sharing permissions for a specific activity map.

Operation details

POST /activitymaps

Specify the following parameters.

body: ***Object***

The activity map properties.

name: ***String***

The friendly name for the activity map.

short_code: ***String***

(Optional) The unique short code that is global to all activity maps.

description: ***String***

The description for the activity map.

weighting: ***String***

(Optional) The metric value that determines how activity is weighted between devices. Supported element values are "bytes", "connections", and "turns".

mode: ***String***

(Optional) The layout of the activity map. Supported values are "2dforce" and "3dforce".

show_alert_status: ***Boolean***

(Optional) Indicates whether to show the alert status for devices on the activity map. If enabled, the color of each device on the map represents the most severe alert level associated with the device.

walks: ***Array of Objects***

The list of one or more walk objects. A walk is the path of traffic composed of one or more steps. Each walk begins with one or more origin devices and expands to connections to peer devices that are based on protocol activity. Each expansion from the origin is a step. The contents of the object are defined in the "walk" section below.

origins: ***Array of Objects***

The list of one or more origin devices of the first step within the walk. Object contents are defined in the "source_object" section below.

object_type: ***String***

The metric source type.

The following values are valid:

- `device`
- `device_group`

object_id: ***Number***

The unique identifier for the source object.

steps: ***Array of Objects***

The list of one or more steps within the walk. Each step is defined by the protocol activity between devices of the previous step to a new set of peer devices. Object contents are defined in the "step" section below.

relationships: ***Array of Objects***

(Optional) The list of one or more filters that define the relationship between two devices. The filters specify which roles and protocols to search for when locating peer devices in the step. Relationships are represented as an edge in the activity map. Object contents are defined in the "relationship" section below. If no value is specified, the operation will locate all peers.

protocol: ***String***

(Optional) The metric protocol associated with the relationship, such as "HTTP" or "DNS". The operation only locates connections between devices over the specified protocol.

role: ***String***

(Optional) The device role associated with the metric protocol of the relationship. The operation only locates connections between devices over the associated protocol in the specified role. Supported role values are "client", "server", or "any". Set to "any" to locate all client, server, and peer device relationships associated with the specified protocol.

peer_in: ***Array of Objects***

(Optional) The list of one or more peer device objects to include in the activity map. Only relationships to peers of the specified source object are included. Object contents are defined in the "source_object" section below.

object_type: ***String***

The metric source type.

The following values are valid:

- `device`
- `device_group`

object_id: ***Number***

The unique identifier for the source object.

peer_not_in: ***Array of Objects***

(Optional) The list of one or more peer device objects to exclude from the activity map. Relationships to peers of the specified source object are excluded. Object contents are defined in the "source_object" section below.

object_type: ***String***

The metric source type.

The following values are valid:

- `device`
- `device_group`

object_id: ***Number***

The unique identifier for the source object.

Specify the body parameter in the following JSON format.

```
{
    "description": "string",
    "mode": "string",
    "name": "string",
    "short_code": "string",
    "show_alert_status": true,
    "walks": {
        "origins": {
            "object_type": "string",
            "object_id": 0
        },
        "steps": {
            "relationships": {
                "protocol": "string",
                "role": "string"
            },
            "peer_in": {
                "object_type": "string",
                "object_id": 0
            },
            "peer_not_in": {
                "object_type": "string",
```

```
                "object_id": 0
            }
        }
    },
    "weighting": "string"
}
```

POST /activitymaps/query

Specify the following parameters.

body: ***Object***

The topology query properties.

from: ***Number***

The beginning timestamp of the time range the query will search, expressed in milliseconds since the epoch.

until: ***Number***

(Optional) The ending timestamp of the time range the query will search, expressed in milliseconds since the epoch. If no value is set, the query end defaults to "now".

weighting: ***String***

(Optional) The metric value that determines how activity is weighted between devices.

The following values are valid:

- `bytes`
- `connections`
- `turns`

edge_annotations: ***Array of Strings***

(Optional) The list of one or more edge annotations to include in the topology query.

The following values are valid:

- `protocols`
- `appearances`

walks: ***Array of Objects***

The list of one or more walk objects to include in the topology query. A walk is the path of traffic composed of one or more steps. Each walk begins with one or more origin devices and expands to connections to peer devices that are based on protocol activity. Each expansion from the origin is a step. Object contents are defined in the "topology_walk" section below.

origins: ***Array of Objects***

The list of one or more origin devices of the first step within the walk. Object contents are defined in the "topology_source" section below.

object_type: ***String***

The type of source object.

The following values are valid:

- `all_devices`
- `device_group`
- `device`

object_id: ***Number***

The unique identifier for the source object. Set to 0 if the value of the "object_type" parameter is "all_devices".

steps: ***Array of Objects***

The list of one or more steps within the walk. Each step is defined by the protocol activity between devices of the previous step to a new set of peer devices. Object coontents are defined in the "topology_step" section below.

relationships: ***Array of Objects***

(Optional) The list of one or more filters that define the relationship between two devices. The filters specify which roles and protocols to search for when locating peer devices in the step. Relationships are represented as an edge in the activity map. If no value is set, the operation includes all peers. Object contents are defined in the "topology_relationship" section below.

role: ***String***

(Optional) The role of the peer device in relation to the origin device.

The following values are valid:

- `client`
- `server`
- `any`

protocol: ***String***

(Optional) The protocol over which the origin device is communicating, such as "HTTP". If no value is set, the object includes any protocol.

peer_in: ***Array of Objects***

(Optional) The list of one or more peer devices to include in the topology graph. Only relationships to peers of the specified source object are included. Object contents are defined in the "topology_source" section below.

object_type: ***String***

The type of source object.

The following values are valid:

- `all_devices`
- `device_group`
- `device`

object_id: ***Number***

The unique identifier for the source object. Set to 0 if the value of the "object_type" parameter is "all_devices".

peer_not_in: ***Array of Objects***

(Optional) The list of one or more peer devices to exclude from the topology graph. Relationships to peer devices of the specified source object are excluded. Object contents are defined in the "topology_source" section below.

object_type: ***String***

The type of source object.

The following values are valid:

- `all_devices`
- `device_group`
- `device`

object_id: ***Number***

The unique identifier for the source object. Set to 0 if the value of the "object_type" parameter is "all_devices".

Specify the body parameter in the following JSON format.

```
{
    "edge_annotations": [],
    "from": 0,
    "until": 0,
    "walks": {
        "origins": {
            "object_type": "string",
            "object_id": 0
        },
        "steps": {
            "relationships": {
                "role": "string",
                "protocol": "string"
            },
            "peer_in": {
                "object_type": "string",
                "object_id": 0
            },
            "peer_not_in": {
                "object_type": "string",
                "object_id": 0
            }
        }
    },
    "weighting": "string"
}
```

GET /activitymaps

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "id": 0,
    "mod_time": 0,
    "mode": "string",
    "name": "string",
    "owner": "string",
    "rights": [
        "string"
    ],
    "short_code": "string",
    "show_alert_status": true,
    "walks": [],
    "weighting": "string"
}
```

GET /activitymaps/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the activity map.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
```

```
    "id": 0,
    "mod_time": 0,
    "mode": "string",
    "name": "string",
    "owner": "string",
    "rights": [
        "string"
    ],
    "short_code": "string",
    "show_alert_status": true,
    "walks": [],
    "weighting": "string"
}
```

POST /activitymaps/{id}/query

Specify the following parameters.

id: ***Number***

The unique identifier for the activity map.

body: ***Object***

The topology query properties.

from: ***Number***

The beginning timestamp of the time range the query will search, expressed in milliseconds since the epoch.

until: ***Number***

(Optional) The ending timestamp of the time range the query will search, expressed in milliseconds since the epoch. If no value is set, the query end defaults to "now".

edge_annotations: ***Array of Strings***

(Optional) The list of one or more edge annotations to include in the topology query.

The following values are valid:

- `protocols`
- `appearances`

Specify the body parameter in the following JSON format.

```
{
    "edge_annotations": [],
    "from": 0,
    "until": 0
}
```

DELETE /activitymaps/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the activity map.

PATCH /activitymaps/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the activity map.

body: ***Object***

The activity map properties to update.

GET /activitymaps/{id}/sharing

Specify the following parameters.

id: ***Number***

The unique identifier for the activity map.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "anyone": "string",
    "groups": {},
    "users": {}
}
```

PUT /activitymaps/{id}/sharing

Specify the following parameters.

body: ***Object***

The users and their permission levels.

id: ***Number***

The unique identifier for the activity map.

PATCH /activitymaps/{id}/sharing

Specify the following parameters.

body: ***Object***

The users and their permission levels.

id: ***Number***

The unique identifier for the activity map.

Alert

Alerts are system notifications that are generated upon specified alert criteria. Default alerts are available in the system, or you can create a custom alert.

Detections and threshold alerts can be set to alert you if a metric crosses the value defined in the alert configuration. Trend alerts cannot be configured through the REST API. For more information, see Alerts.

Note: Machine learning detections require a connection to ExtraHop Cloud Services.

The following table displays all of the operations you can perform this resource:

Operation	Description
GET /alerts	Retrieve all alerts.
POST /alerts	Create a new alert with specified values.
DELETE /alerts/{id}	Delete a specific alert.
GET /alerts/{id}	Retrieve a specific alert.

Operation	Description
PATCH /alerts/{id}	Apply updates to a specific alert.
GET /alerts/{id}/applications	Retrieve all applications that have a specific alert assigned.
POST /alerts/{id}/applications	Assign and unassign a specific alert to applications.
DELETE /alerts/{id}/applications/{child-id}	Unassign an application from a specific alert.
POST /alerts/{id}/applications/{child-id}	Assign an application to a specific alert.
GET /alerts/{id}/devicegroups	Retrieve all device groups that have a specific alert assigned.
POST /alerts/{id}/devicegroups	Assign and unassign a specific alert to device groups.
DELETE /alerts/{id}/devicegroups/{child-id}	Unassign a device group from a specific alert.
POST /alerts/{id}/devicegroups/{child-id}	Assign a device group to a specific alert.
GET /alerts/{id}/devices	Retrieve all devices that have a specific alert assigned.
POST /alerts/{id}/devices	Assign and unassign a specific alert to devices.
DELETE /alerts/{id}/devices/{child-id}	Unassign a device from a specific alert.
POST /alerts/{id}/devices/{child-id}	Assign a device to a specific alert.
GET /alerts/{id}/emailgroups	Retrieve all email groups that have a specific alert assigned.
POST /alerts/{id}/emailgroups	Assign and unassign a specific alert to email groups.
DELETE /alerts/{id}/emailgroups/{child-id}	Unassign a email group from a specific alert.
POST /alerts/{id}/emailgroups/{child-id}	Assign a email group to a specific alert.
GET /alerts/{id}/exclusionintervals	Retrieve all exclustion intervals that have a specific alert assigned.
POST /alerts/{id}/exclusionintervals	Assign and unassign a specific alert to exclustion intervals.
DELETE /alerts/{id}/exclusionintervals/{child-id}	Unassign an exclusion interval from a specific alert.
POST /alerts/{id}/exclusionintervals/{child-id}	Assign an exclusion interval to a specific alert.
GET /alerts/{id}/networks	Retrieve all networks that have a specific alert assigned.
POST /alerts/{id}/networks	Assign and unassign a specific alert to networks.
DELETE /alerts/{id}/networks/{child-id}	Unassign a network from a specific alert.
POST /alerts/{id}/networks/{child-id}	Assign a network to a specific alert.
GET /alerts/{id}/stats	Retrieve all additional statistics for a specific alert.

Operation details

GET /alerts

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "apply_all": true,
    "author": "string",
    "categories": [
        "string"
    ],
    "cc": [],
    "description": "string",
    "disabled": true,
    "field_name": "string",
    "field_name2": "string",
    "field_op": "string",
    "id": 0,
    "interval_length": 0,
    "mod_time": 0,
    "name": "string",
    "notify_snmp": true,
    "object_type": "string",
    "operand": "string",
    "operator": "string",
    "param": {},
    "param2": {},
    "protocols": [
        "string"
    ],
    "refire_interval": 0,
    "severity": 0,
    "stat_name": "string",
    "type": "string",
    "units": "string"
}
```

POST /alerts

Specify the following parameters.

body: ***Object***

Apply the specified property values to the new alert.

description: ***String***

An optional description for the alert.

notify_snmp: ***Boolean***

Indicates whether to send an SNMP trap when an alert is generated.

field_op: ***String***

The type of comparison between the field_name and field_name2 fields when applying a ratio. Only applicable to threshold alerts.

The following values are valid:

- `/`
- `null`

stat_name: ***String***

The statistic name for the alert. Only applicable to threshold alerts.

disabled: ***Boolean***

Indicates whether the alert is disabled.

operator: ***String***

The logical operator applied when comparing the value of the operand field to alert conditions. Only applicable to threshold alerts.

The following values are valid:

- `==`
- `>`
- `<`
- `>=`
- `<=`

operand: ***String***

The value to compare against alert conditions. The compare method is specified by the value of the operator field. Only applicable to threshold alerts.

field_name: ***String***

The name of the monitored metric. Only applicable to threshold alerts.

name: ***String***

The unique, friendly name for the alert.

cc: ***Array of Strings***

The list of email addresses, not included in an email group, to receive notifications.

apply_all: ***Boolean***

Indicates whether the alert is assigned to all available data sources.

severity: ***Number***

The severity level of the alert, which is displayed in the Alert History, email notifications, and SNMP traps. Severity levels 0-2 require immediate attention. Severity levels are described in the REST API Guide.

author: ***String***

The name of the user that created the alert.

param: ***Object***

The first alert parameter, which is either a key pattern or a data point. Only applicable to threshold alerts.

interval_length: ***Number***

The length of the alert interval, expressed in seconds. Only applicable to threshold alerts.

The following values are valid:

- `30`
- `60`
- `120`
- `300`
- `600`
- `900`
- `1200`
- `1800`

param2: ***Object***

The second alert parameter, which is either a key pattern or a data point. Only applicable to threshold alerts.

units: ***String***

The interval in which to evaluate the alert condition. Only applicable to threshold alerts.

The following values are valid:

- `none`
- `period`
- `1 sec`
- `1 min`
- `1 hr`

field_name2: ***String***

The second monitored metric when applying a ratio. Only applicable to threshold alerts.

refire_interval: ***Number***

The time interval in which alert conditions are monitored, expressed in seconds.

The following values are valid:

- `300`
- `600`
- `900`
- `1800`
- `3600`
- `7200`
- `14400`

type: ***String***

The type of alert.

The following values are valid:

- `detection`
- `threshold`

object_type: ***String***

The type of metric source monitored by the alert configuration. Only applicable to detection alerts.

The following values are valid:

- `application`
- `device`

protocols: ***Array of Strings***

(Optional) The list of monitored protocols. Only applicable to detection alerts.

categories: ***Array of Strings***

(Optional) The list of one or more detection categories. An alert is generated only if a detection is identified in the specified categories. Only applicable to detection alerts.

Specify the body parameter in the following JSON format.

```
{
    "apply_all": true,
    "author": "string",
    "categories": [
        "string"
    ],
    "cc": [],
    "description": "string",
    "disabled": true,
    "field_name": "string",
```

```
    "field_name2": "string",
    "field_op": "string",
    "interval_length": 0,
    "name": "string",
    "notify_snmp": true,
    "object_type": "string",
    "operand": "string",
    "operator": "string",
    "param": {},
    "param2": {},
    "protocols": [
        "string"
    ],
    "refire_interval": 0,
    "severity": 0,
    "stat_name": "string",
    "type": "string",
    "units": "string"
}
```

GET /alerts/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the alert.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "apply_all": true,
    "author": "string",
    "categories": [
        "string"
    ],
    "cc": [],
    "description": "string",
    "disabled": true,
    "field_name": "string",
    "field_name2": "string",
    "field_op": "string",
    "id": 0,
    "interval_length": 0,
    "mod_time": 0,
    "name": "string",
    "notify_snmp": true,
    "object_type": "string",
    "operand": "string",
    "operator": "string",
    "param": {},
    "param2": {},
    "protocols": [
        "string"
    ],
    "refire_interval": 0,
    "severity": 0,
    "stat_name": "string",
    "type": "string",
    "units": "string"
}
```

DELETE /alerts/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the alert.

PATCH /alerts/{id}

Specify the following parameters.

body: ***Object***
Apply the specified property value updates to the alert.

id: ***Number***
The unique identifier for the alert.

GET /alerts/{id}/stats

Specify the following parameters.

id: ***Number***
The unique identifier for the alert.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "alert_id": 0,
    "field_name": "string",
    "id": 0,
    "param": "string",
    "stat_name": "string"
}
```

GET /alerts/{id}/devicegroups

Specify the following parameters.

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/devicegroups

Specify the following parameters.

body: ***Object***
The list of unique identifiers for device groups that is assigned and unassigned to the alert.

assign: ***Array of Numbers***
IDs of resources to assign

unassign: ***Array of Numbers***
IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/devicegroups/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the device group.
id: ***Number***
The unique identifier for the alert.

DELETE /alerts/{id}/devicegroups/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the device group.
id: ***Number***
The unique identifier for the alert.

GET /alerts/{id}/emailgroups

Specify the following parameters.

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/emailgroups

Specify the following parameters.

body: ***Object***
The list of unique identifiers for email groups that is assigned and unassigned to the alert.
assign: ***Array of Numbers***
IDs of resources to assign
unassign: ***Array of Numbers***
IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/emailgroups/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the email group.

id: ***Number***
The unique identifier for the alert.

DELETE /alerts/{id}/emailgroups/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the email group.
id: ***Number***
The unique identifier for the alert.

GET /alerts/{id}/exclusionintervals

Specify the following parameters.

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/exclusionintervals

Specify the following parameters.

body: ***Object***
The list of unique identifiers for exclusion intervals that is assigned and unassigned to the alert.
assign: ***Array of Numbers***
IDs of resources to assign
unassign: ***Array of Numbers***
IDs of resources to unassign
Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/exclusionintervals/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the exclusion interval.
id: ***Number***
The unique identifier for the alert.

DELETE /alerts/{id}/exclusionintervals/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the exclusion interval.

id: ***Number***
The unique identifier for the alert.

GET /alerts/{id}/devices

Specify the following parameters.

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/devices

Specify the following parameters.

body: ***Object***
The list of unique identifiers for devices that is assigned and unassigned to the alert.
assign: ***Array of Numbers***
IDs of resources to assign
unassign: ***Array of Numbers***
IDs of resources to unassign
Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the device.
id: ***Number***
The unique identifier for the alert.

DELETE /alerts/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the device.
id: ***Number***
The unique identifier for the alert.

GET /alerts/{id}/networks

Specify the following parameters.

id: ***Number***
The unique identifier for the alert.

POST /alerts/{id}/networks

Specify the following parameters.

body: ***Object***

The list of unique identifiers for networks that is assigned and unassigned to the alert.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the alert.

POST /alerts/{id}/networks/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the network.

id: ***Number***

The unique identifier for the alert.

DELETE /alerts/{id}/networks/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the network.

id: ***Number***

The unique identifier for the alert.

GET /alerts/{id}/applications

Specify the following parameters.

id: ***Number***

The unique identifier for the alert.

POST /alerts/{id}/applications

Specify the following parameters.

body: ***Object***

The list of unique identifiers for applications that is assigned and unassigned to the alert.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the alert.

POST /alerts/{id}/applications/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the application.

id: ***Number***

The unique identifier for the alert.

DELETE /alerts/{id}/applications/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the application.

id: ***Number***

The unique identifier for the alert.

Alert severity levels

The severity level you specify for an alert is displayed on the Alerts page, email notifications, and SNMP traps.

The following severity levels are supported. Severity levels 0-2 require immediate attention.

Value	Name	Description
0	Emergency	System functionality is unavailable.
1	Alert	Immediate action is required.
2	Critical	Critical conditions are occurring.
3	Error	Error conditions are occurring.
4	Warning	Warning conditions are occurring.
5	Notice	Normal operations are occurring with significant conditions, such as a restart.
6	Info	Normal operations are occurring with process updates.
7	Debug	Debug-level messages are available.

Analysis Priority

The ExtraHop system analyzes and classifies traffic for every device it discovers. Your license reserves capacity for the ExtraHop system to collect metrics for high-value devices. This capacity is associated with two analysis levels: Advanced Analysis and Standard Analysis.

You can specify which devices receive Advanced Analysis and Standard Analysis levels by configuring analysis priority rules. Analysis priorities help inform the ExtraHop system about which devices are important in your environment. A third analysis level, Discovery Mode, is available for devices that are not in Advanced or Standard Analysis.

Note: By default, each sensor manages its own analysis priorities. If the sensor is connected to a console, you can centrally manage these shared system settings from the console.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /analysispriority/config/{sensor_id}	Retrieve the analysis priority rules for a specific sensor.
PUT /analysispriority/config/{sensor_id}	Replace the analysis priority rules for a specific sensor.
GET /analysispriority/{sensor_id}/manager	Retrieve the system that is configured to manage the analysis priority rules for the sensor.
PATCH /analysispriority/{sensor_id}/manager	Update the system that manages analysis priority rules for a specific sensor.

Operation details

GET /analysispriority/{appliance_id}/manager

Specify the following parameters.

appliance_id: ***Number***

The identifier for the local sensor. This value must be set to 0.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "manager": {}
}
```

GET /analysispriority/config/{appliance_id}

Specify the following parameters.

appliance_id: ***Number***

The identifier for a sensor. Set this value to 0 if calling on a sensor.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "advanced_rules": [],
    "autofill_advanced": true,
    "autofill_standard": true,
    "is_in_effect": true,
    "standard_rules": []
```

```
}
```

PUT /analysispriority/config/{appliance_id}

Specify the following parameters.

body: ***Object***

The properties of the priority analysis rules.

autofill_advanced: ***Boolean***

Indicates whether to automatically place devices in Advanced Analysis until capacity is reached. Devices in the advanced_rules list are prioritized, followed by devices in the standard_rules list, and then by the discovery time for the device. The capacity for Advanced Analysis is determined by the ExtraHop system license.

advanced_rules: ***Array of Objects***

(Optional) The Advanced Analysis priority rules for a device group.

type: ***String***

The type of group the analysis priority rules apply to.

The following values are valid:

- `device_group`

object_id: ***Number***

The unique identifier for the group.

description: ***String***

(Optional) The description for analysis priority rules.

autofill_standard: ***Boolean***

Indicates whether to automatically place devices in Standard Analysis until total capacity is reached. Devices in the standard_rules list are prioritized, followed by the discovery time for the device. The total capacity is determined by the ExtraHop system license.

standard_rules: ***Array of Objects***

(Optional) The Standard Analysis priority rules for a device group.

type: ***String***

The type of group the analysis priority rules apply to.

The following values are valid:

- `device_group`

object_id: ***Number***

The unique identifier for the group.

description: ***String***

(Optional) The description for analysis priority rules.

Specify the body parameter in the following JSON format.

```
{
    "advanced_rules": {
        "type": "string",
        "object_id": 0,
        "description": "string"
    },
    "autofill_advanced": true,
    "autofill_standard": true,
    "standard_rules": {
        "type": "string",
        "object_id": 0,
```

```
        "description": "string"
    }
}
```

appliance_id: ***Number***

The identifier for a sensor. Set this value to 0 if calling on a sensor.

PATCH /analysispriority/{appliance_id}/manager

Specify the following parameters.

body: ***Object***

The ID of the sensor or console that will manage analysis priority rules for the local sensor. Set this value to 0 to restore management to the local sensor.

manager: ***Number***

The unique identifier for the managing sensor or console.

Specify the body parameter in the following JSON format.

```
{
    "manager": 0
}
```

appliance_id: ***Number***

The identifier for the local sensor. This value must be set to 0.

APIKey

An API key enables a user to perform operations through the ExtraHop REST API.

You can generate the initial API key for the setup user account through the REST API. All other API keys are generated through the API Access page in the Administration settings.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /apikeys	Retrieve all API keys.
POST /apikeys	Create the initial API key for the setup user account.
GET /apikeys/{keyid}	Retrieve information about a specific API key.

Operation details

GET /apikeys

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "id": 0,
    "key": "string",
    "time_added": 0,
    "user_id": 0,
    "username": "string"
}
```

GET /apikeys/{keyid}

Specify the following parameters.

keyid: ***Number***

The unique identifier for the API key.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "id": 0,
    "key": "string",
    "time_added": 0,
    "user_id": 0,
    "username": "string"
}
```

POST /apikeys

Specify the following parameters.

body: ***Object***

The password of the setup user.

password: ***String***

The password for the setup user.

Specify the body parameter in the following JSON format.

```
{
    "password": "string"
}
```

Appliance

The ExtraHop system consists of a network of connected ExtraHop appliances, such as sensors, consoles, recordstores, and packetstores that perform tasks such as monitoring traffic, analyzing data, storing data, and identifying detections.

You can retrieve information and establish connections for local and remote ExtraHop appliances.

Note: You can only establish a connection between similar ExtraHop appliances, such as Reveal(x) Enterprise or Performance.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /appliances	Retrieve all remote ExtraHop appliances connected to the local appliance.
POST /appliances	Establish a new connection to a remote ExtraHop appliance.
DELETE /appliances/{id}	Disconnect a specific ExtraHop appliance from this console.
GET /appliances/{id}	Retrieve a specific remote ExtraHop appliance connected to the local appliance (only valid on consoles).

Operation	Description
GET /appliances/{id}/cloudservices	Retrieve the status of ExtraHop Cloud Services on this appliance (only valid on consoles).
GET /appliances/{id}/productkey	Retrieve the product key for a specified appliance (only valid on consoles).
GET /appliances/firmware/next	Retrieve firmware versions that remote ExtraHop systems can be upgraded to (only valid on consoles).
POST /appliances/firmware/upgrade	Upgrade firmware on remote ExtraHop systems connected to the local system. Firmware images are downloaded from ExtraHop Cloud Services (only valid on consoles).

Operation details

GET /appliances

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "add_time": 0,
    "advanced_analysis_capacity": 0,
    "analysis_levels_managed": true,
    "connection_type": "string",
    "data_access": true,
    "display_name": "string",
    "fingerprint": "string",
    "firmware_version": "string",
    "hostname": "string",
    "id": 0,
    "license_status": "string",
    "licensed_features": {},
    "managed_by_local": true,
    "manages_local": true,
    "nickname": "string",
    "platform": "string",
    "status_message": "string",
    "sync_time": 0,
    "total_capacity": 0,
    "uuid": "string"
}
```

GET /appliances/{id}

Specify the following parameters.

id: ***Number***

Specify the unique identifier for the remote appliance.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "add_time": 0,
    "advanced_analysis_capacity": 0,
    "analysis_levels_managed": true,
    "connection_type": "string",
```

```
    "data_access": true,
    "display_name": "string",
    "fingerprint": "string",
    "firmware_version": "string",
    "hostname": "string",
    "id": 0,
    "license_status": "string",
    "licensed_features": {},
    "managed_by_local": true,
    "manages_local": true,
    "nickname": "string",
    "platform": "string",
    "status_message": "string",
    "sync_time": 0,
    "total_capacity": 0,
    "uuid": "string"
}
```

DELETE /appliances/{id}

Specify the following parameters.

id: ***Number***

Specify the unique identifier for the remote appliance.

POST /appliances

Specify the following parameters.

body: ***Object***

Specify the properties of the new connection.

host: ***String***

The hostname of the remote appliance.

remote_setup_password: ***String***

(Optional) The password for the setup user account on the target EXA or ExtraHop packetstore. This parameter is not required if the remote appliance is a node in an Explore cluster already connected to the console. This parameter is not valid if the remote appliance is a sensor.

remote_pairing_token: ***String***

(Optional) The token generated on the target sensor. You must specify this parameter to authenticate to the target sensor. This parameter is not valid if you are connecting to an EXA or ExtraHop packetstore.

fingerprint: ***String***

(Optional) The fingerprint of the remote appliance. If you are connecting a console to an EXA or ExtraHop packetsore, this field is required. Otherwise, to bypass fingerprint verification, specify 'insecure_skip_verification'. Note that bypassing verification can allow for man-in-the-middle attacks.

reset_configuration: ***Boolean***

(Optional) Indicates whether to reset the configuration of the remote appliance.

remote_nickname_for_local: ***String***

(Optional) The nickname for the remote appliance, referred to by the local appliance. If you are connecting a sensor to any other appliance, this field is required.

local_nickname_for_remote: ***String***

(Optional) The nickname for the local appliance, referred to by the remote appliance.

remote_appliance_type: ***String***

The type of appliance for the new connection.

The following values are valid:

- `command`
- `explore`
- `discover`
- `trace`

manages_local: ***Boolean***

(Optional) Indicates whether the remote appliance manages the local appliance.

managed_by_local: ***Boolean***

Indicates whether the remote appliance is managed by the local appliance. If you are connecting a console to a sensor, this field is not required because console always manage connected sensors.

data_access: ***Boolean***

Indicates whether data can be shared between the local and remote appliances.

product_key: ***String***

(Optional) The product key for the target appliance. If this parameter is specified, the target appliance is licensed with the product key. This parameter is invalid when the remote_pairing_token parameter is specified.

Specify the body parameter in the following JSON format.

```
{
    "data_access": true,
    "fingerprint": "string",
    "host": "string",
    "local_nickname_for_remote": "string",
    "managed_by_local": true,
    "manages_local": true,
    "product_key": "string",
    "remote_appliance_type": "string",
    "remote_nickname_for_local": "string",
    "remote_pairing_token": "string",
    "remote_setup_password": "string",
    "reset_configuration": true
}
```

GET /appliances/{id}/productkey

Specify the following parameters.

id: ***Number***

Specify the unique identifier for the remote appliance.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "product_key": "string"
}
```

GET /appliances/firmware/next

Specify the following parameters.

ids: ***String***

(Optional) A CSV list of unique identifiers for the remote appliances. If this parameter is specified, the operation returns firmware versions that any of the specified remote appliances can be upgraded to. If this parameter is not specified, the operation returns firmware versions that any remote appliance can be upgraded to.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "release": "string",
    "versions": []
}
```

GET /appliances/{id}/cloudservices

Specify the following parameters.

id: ***Number***

Specify the unique identifier for the appliance. This value must be set to 0, which selects the local appliance.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "connection_status": "string",
    "connection_status_color": "string",
    "enabled_services": [],
    "last_active_time": 0,
    "last_analyzed_time": 0
}
```

POST /appliances/firmware/upgrade

Specify the following parameters.

body: ***Object***

The firmware upgrade options.

version: ***String***

The firmware version to upgrade appliances to. You can retrieve a list of valid versions with the GET /api/v1/appliances/firmware/next operation.

system_ids: ***Array of Numbers***

A list of unique identifiers for the remote appliances. You can retrieve appliance IDs with the GET /api/v1/appliances operation; appliance IDs are returned in the id fields of the response.

Specify the body parameter in the following JSON format.

```
{
    "system_ids": [],
    "version": "string"
}
```

Application

Applications are user-defined groups that collect metrics identified through triggers across multiple types of traffic. The default All Activity application contains all collected metrics.

The following table displays all of the operations you can perform on the application resource:

Operation	Description
GET /applications	Retrieve all applications that were active within a specific timeframe.
POST /applications	Create a new application.
GET /applications/{id}	Retreive a specific application.
PATCH /applications/{id}	Update a specific application.
GET /applications/{id}/activity	Retrieve all activity for a specific application.
GET /applications/{id}/alerts	Retrieve all alerts that are assigned to a specific application.
POST /applications/{id}/alerts	Assign and unassign alerts to a specific application.
DELETE /applications/{id}/alerts/{child-id}	Unassign an alert from a specific application.
POST /applications/{id}/alerts/{child-id}	Assign an alert to a specific application.
GET /applications/{id}/dashboards	Retrieve all dashboards related to a specific application.

Operation details

GET /applications/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the application.

include_criteria: ***Boolean***

(Optional) Indicates whether to include the criteria associated with the application in the response.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "criteria": [],
    "description": "string",
    "discovery_id": "string",
    "display_name": "string",
    "extrahop_id": "string",
    "id": 0,
    "mod_time": 0,
    "node_id": 0,
    "user_mod_time": 0
}
```

POST /applications

Specify the following parameters.

body: ***Object***

The properties of the application.

node_id: ***Number***

(Optional) The unique identifier for the sensor that this application is associated with. The identifier can be retrieved through the GET /nodes operation. This field is valid only on a console.

discovery_id: ***String***

The unique identifier for the application, which is displayed on the application page in the ExtraHop system.

display_name: ***String***

The friendly name for the application.

description: ***String***

(Optional) An optional description for the application.

criteria: ***Array of Objects***

(Optional) An array of protocol and source criteria associated with the application. The contents of this array are defined in the 'criteria' section below.

protocol_default: ***String***

The default protocols monitored by the application. Supported values are 'any' and 'none'.

sources: ***Array of Objects***

An array containing one or more metric sources associated with the application. The application only collects metrics from the specified sources. The contents of this array are defined in the 'source' section below.

type: ***String***

The type of metric source associated with the application. Supported source type values are 'device' and 'device_group'.

id: ***Number***

The unique identifier for the device or device group associated with the application.

protocols: ***Object***

(Optional) The list of one or more protocol and role mappings associated with the application. The application only collects metrics from the specified protocols. The format of each protocol is {'protocol':'role'}. Example: {'http': 'server'}. Supported role values are 'client', 'server', 'any', or 'none'.

Specify the body parameter in the following JSON format.

```
{
    "criteria": {
        "protocol_default": "string",
        "sources": {
            "type": "string",
            "id": 0
        },
        "protocols": {}
    },
    "description": "string",
    "discovery_id": "string",
    "display_name": "string",
    "node_id": 0
}
```

PATCH /applications/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property updates to the application.

id: ***Number***

The unique identifier for the application.

GET /applications

Specify the following parameters.

active_from: ***Number***

(Optional) Return only applications that are active after the specified time. Positive values specify the time in milliseconds since the epoch. Negative values specify the time in milliseconds before the current time.

active_until: ***Number***

(Optional) Return only applications that are active before the specified time. Positive values specify the time in milliseconds since the epoch. Negative values specify the time in milliseconds before the current time.

limit: ***Number***

(Optional) Limit the number of applications that are returned to the specified maximum number.

offset: ***Number***

(Optional) Skip the first n application results. This parameter is often combined with the limit parameter.

search_type: ***String***

The object type to search for.

The following values are valid:

- `any`
- `name`
- `node`
- `discovery_id`
- `extrahop-id`

value: ***String***

(Optional) The search criteria. Add a forward slash before and after the criteria to apply RegEx matching.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "criteria": [],
    "description": "string",
    "discovery_id": "string",
    "display_name": "string",
    "extrahop_id": "string",
    "id": 0,
    "mod_time": 0,
    "node_id": 0,
    "user_mod_time": 0
}
```

GET /applications/{id}/activity

Specify the following parameters.

id: ***Number***

The unique identifier for the application.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "application_id": 0,
    "from_time": 0,
    "id": 0,
    "mod_time": 0,
    "stat_name": "string",
    "until_time": 0
}
```

GET /applications/{id}/alerts

Specify the following parameters.

id: ***Number***

Retrieve the unique identifier for the application.

direct_assignments_only: ***Boolean***

(Optional) Indicates whether results are restricted to alerts that are directly assigned to the application.

POST /applications/{id}/alerts

Specify the following parameters.

body: ***Object***

Assign or unassign the specified list of unique identifiers for alerts.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

Provide a unique identifier for the application.

POST /applications/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the alert.

id: ***Number***

The unique identifier for the application.

DELETE /applications/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the alert.

id: ***Number***

The unique identifier for the application.

GET /applications/{id}/dashboards

Specify the following parameters.

id: ***Number***

The unique identifier for the application.

Audit log

The audit log displays a record of all recorded system administration and configuration activity, such as the time of the activity, the user who performed the activity, the operation, operation details, and system component..

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /auditlog	Retrieve all audit log messages.

Operation details

GET /auditlog

Specify the following parameters.

limit: ***Number***

(Optional) The maximum number of log messages to return.

offset: ***Number***

(Optional) The number of log messages to skip in the results. Returns log messages starting from the offset value.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "body": {},
    "id": 0,
    "occur_time": 0,
    "time": 0
}
```

Auth

You can configure secure, single sign-on (SSO) authentication to the ExtraHop system through one or more security assertion markup language (SAML) identity providers.

When a user logs in to an ExtraHop system that is configured as a service provider (SP) for SAML SSO authentication, the ExtraHop system requests authorization from the appropriate identity provider (IdP).

The identity provider authenticates the user's credentials and then returns the authorization for the user to the ExtraHop system. The user is then able to access the ExtraHop system.

Operation	Description
GET /auth/identityproviders	Retrieve all identity providers.
POST /auth/identityproviders	Add an identity provider for remote authentication.
DELETE /auth/identityproviders/{id}	Delete a specific identity provider.
GET /auth/identityproviders/{id}	Retrieve a specific identity provider.
PATCH /auth/identityproviders/{id}	Update an existing identity provider.
GET /auth/identityproviders/{id}/privileges	Retrieve the privilege settings for a specific identity provider.
PATCH /auth/identityproviders/{id}/privileges	Update the privilege settings for a specific identity provider.
GET /auth/samlsp	Retrieve SAML security provider (SP) metadata for this ExtraHop system.

Operation details

POST /auth/identityproviders

Specify the following parameters.

body: ***Object***

Parameters for the identity provider.

name: ***String***

The name of the identity provider.

enabled: ***Boolean***

Indicates whether authentication through the identity provider is enabled on the ExtraHop system.

entity_id: ***String***

(Optional) The SAML 2.0 entityID.

sso_url: ***String***

(Optional) The SAML 2.0 Single Sign-On (SSO) URL.

signing_certificate: ***String***

(Optional) The SAML 2.0 X.509 signing certificate in PEM format.

type: ***String***

The type of identity provider.

The following values are valid:

- `saml`

auto_provision_users: ***Boolean***

Indicates whether a user can be created on the ExtraHop system from the identity provider.

Specify the body parameter in the following JSON format.

```
{
    "auto_provision_users": true,
    "enabled": true,
    "entity_id": "string",
```

```
    "name": "string",
    "signing_certificate": "string",
    "sso_url": "string",
    "type": "string"
}
```

GET /auth/identityproviders

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "auto_provision_users": true,
    "enabled": true,
    "entity_id": "string",
    "id": 0,
    "name": "string",
    "signing_certificate": "string",
    "sso_url": "string",
    "type": "string"
}
```

GET /auth/identityproviders/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the identity provider.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "auto_provision_users": true,
    "enabled": true,
    "entity_id": "string",
    "id": 0,
    "name": "string",
    "signing_certificate": "string",
    "sso_url": "string",
    "type": "string"
}
```

PATCH /auth/identityproviders/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the identity provider.

body: ***Object***

The parameters for the identity provider.

DELETE /auth/identityproviders/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the identity provider.

GET /auth/identityproviders/{id}/privileges

Specify the following parameters.

id: ***Number***

The unique identifier for the identity provider.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "detectionsaccesslevel": {},
    "packetslevel": {},
    "writelevel": {}
}
```

PATCH /auth/identityproviders/{id}/privileges

Specify the following parameters.

id: ***Number***

The unique identifier for the identity provider.

body: ***Object***

An object that contains the privilege settings.

GET /auth/samlsp

Specify the following parameters.

xml: ***Boolean***

(Optional) Indicates whether to retrieve the SAML 2.0 XML metadata.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "acs_url": "string",
    "entity_id": "string",
    "xml": "string"
}
```

Bundle

Bundles are JSON-formatted documents that contain information about selected system configuration, such as triggers, dashboards, applications, or alerts.

You can create a bundle and then transfer those configurations to another ExtraHop system, or save the bundle as a backup. Bundles can also be downloaded from ExtraHop Solution Bundles and applied through the REST API. For more information, see Bundles.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /bundles	Retrieve metadata about all bundles on the ExtraHop system.
POST /bundles	Upload a new bundle to the ExtraHop system.
DELETE /bundles/{id}	Delete a specific bundle.

Operation	Description
GET /bundles/{id}	Retrieve a specific bundle export.
POST /bundles/{id}/apply	Apply a saved bundle to the ExtraHop system.

Operation details

GET /bundles

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "built_in": true,
    "created_time": 0,
    "description": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string"
}
```

POST /bundles

Specify the following parameters.

body: ***String***

A JSON formatted bundle export.

name: ***String***

The friendly name for the bundle.

description: ***String***

(Optional) An optional description for the bundle.

Specify the body parameter in the following JSON format.

```
{
    "description": "string",
    "name": "string"
}
```

GET /bundles/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the bundle.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "built_in": true,
    "created_time": 0,
    "description": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string"
}
```

DELETE /bundles/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the bundle.

POST /bundles/{id}/apply

Specify the following parameters.

id: ***Number***
The unique identifier for the bundle.

body: ***Object***
The configuration options for applying the bundle.

policy: ***String***
Indicates whether conflicting objects should be overwritten or skipped.

The following values are valid:

- `overwrite`
- `skip`

include_assignments: ***Boolean***
Indicates whether object assignments should be restored with the bundle.

node_ids: ***Array of Numbers***
A list of unique identifiers for the sensors on which to apply the bundle. This field is valid only on a console.

Specify the body parameter in the following JSON format.

```
{
    "include_assignments": true,
    "node_ids": [],
    "policy": "string"
}
```

Cloud

This resource enables you to connect your on-premises sensors to Reveal(x) 360 For more information, see Connect to Reveal(x) 360 from self-managed sensors.

The following table displays all of the operations you can perform on this resource:

Operation	Description
POST /cloud/connect	Connect the ExtraHop system to Reveal(x) 360.

Operation details

POST /cloud/connect

Specify the following parameters.

body: ***Object***
The token you generated from Reveal(x) 360.

cloud_token: ***String***

The token you generated from Reveal(x) 360.

nickname: ***String***

A nickname to easily identify the sensor.

Specify the body parameter in the following JSON format.

```
{
    "cloud_token": "string",
    "nickname": "string"
}
```

Custom device

You can create a custom device by defining a set of rules.

For example, you can create a custom device that has an IP address on a specified VLAN. By default, all IP addresses outside of the locally-monitored broadcast domains are aggregated behind a router. To identify devices that are behind that router, you can create a custom device, and then collect metrics from the device. For more information, see Create custom devices through the REST API.

Note: The custom device resource is not available on consoles.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /customdevices	Retrieve all custom devices.
POST /customdevices	Create a custom device.
DELETE /customdevices/{id}	Delete a specific custom device.
GET /customdevices/{id}	Retrieve a specific custom device.
PATCH /customdevices/{id}	Update a specific custom device.

Operation details

GET /customdevices

Specify the following parameters.

include_criteria: ***Boolean***

(Optional) Indicates whether the custom device criteria should be included.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "author": "string",
    "criteria": [],
    "description": "string",
    "disabled": true,
    "extrahop_id": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string"
}
```

GET /customdevices/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the custom device.

include_criteria: ***Boolean***

(Optional) Indicates whether the custom device criteria should be included.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "author": "string",
    "criteria": [],
    "description": "string",
    "disabled": true,
    "extrahop_id": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string"
}
```

DELETE /customdevices/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the custom device.

POST /customdevices

Specify the following parameters.

body: ***Object***

Apply the specified property values to the new custom device.

author: ***String***

The name of the custom device creator.

extrahop_id: ***String***

(Optional) A unique identifier for the custom device. If this field is not specified, an ID is generated from the custom device name. The ID cannot contain spaces and cannot be changed after the custom device is saved.

name: ***String***

The friendly name for the custom device.

description: ***String***

(Optional) An optional description of the custom device.

disabled: ***Boolean***

Indicates whether the custom device is inactive.

criteria: ***Array of Objects***

(Optional) An array of custom device criteria for this device. If this field is specified with the PATCH method, all previously specified criteria are deleted.

ipaddr: ***String***

The IP address to match the custom device to.

ipaddr_direction: ***String***

The direction of traffic to match the ipaddr address to. The criteria determines which direction of traffic to or from the ipaddr address is matched.

The following values are valid:

- `any`
- `dst`
- `src`

ipaddr_peer: ***String***

The IP address that the ipadder address is communicating with to match the custom device to. If specified, this parameter limits the traffic matched by the custom device. For example, if ipaddr_direction is "src", the custom device only matches traffic to the ipaddr_peer address from the ipaddr address. This parameter is only valid if ipaddr is specified and ipaddr_direction is not "any".

src_port_min: ***Number***

The lower source port boundary to match the custom device to.

src_port_max: ***Number***

The maximum source port boundary to match the custom device to.

dst_port_min: ***Number***

The lower destination port boundary to match the custom device to.

dst_port_max: ***Number***

The maximum destination port boundary to match the custom device to.

vlan_min: ***Number***

The lower VLAN boundary to match the custom device to.

vlan_max: ***Number***

The maximum VLAN boundary to match the custom device to.

Specify the body parameter in the following JSON format.

```
{
    "author": "string",
    "criteria": {
        "ipaddr": "string",
        "ipaddr_direction": "string",
        "ipaddr_peer": "string",
        "src_port_min": 0,
        "src_port_max": 0,
        "dst_port_min": 0,
        "dst_port_max": 0,
        "vlan_min": 0,
        "vlan_max": 0
    },
    "description": "string",
    "disabled": true,
    "extrahop_id": "string",
    "name": "string"
}
```

PATCH /customdevices/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property value updates to the custom device.

id: ***Number***

The unique identifier for the custom device.

Customization

The Customization resource enables you to manage backups files on the ExtraHop system. You must have unlimited privileges to perform operations on this resource.

Backup files contain both customizations and systems resources. Customizations are user-defined objects, such as alerts, dashboards, triggers, and custom metrics. System resources are items such as bundles, local users and groups, and the SSL certificate. For more information, see Back up and restore a sensor or console.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /customizations	Retrieve all backup files.
POST /customizations	Create a backup file.
GET /customizations/status	Retrieve status details for the most recent backup attempt.
DELETE /customizations/{id}	Delete a specific backup file.
GET /customizations/{id}	Retrieve a specific backup file.
POST /customizations/{id}/apply	Restore only customizations from a specific backup file.
POST /customizations/{id}/download	Download a specific backup file.

Operation details

GET /customizations

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "auto": true,
    "create_time": 0,
    "id": 0,
    "name": "string",
    "recovered": true
}
```

POST /customizations

Specify the following parameters.

body: ***Object***

A unique name for the backup file.

name: ***String***

A unique name for the backup file.

Specify the body parameter in the following JSON format.

```
{
    "name": "string"
}
```

GET /customizations/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the backup file.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "auto": true,
    "create_time": 0,
    "id": 0,
    "name": "string",
    "recovered": true
}
```

DELETE /customizations/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the backup file.

POST /customizations/{id}/apply

Specify the following parameters.

id: ***Number***

The unique identifier for the backup file.

POST /customizations/{id}/download

Specify the following parameters.

id: ***Number***

The unique identifier for the backup file.

GET /customizations/status

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "did_last_succeed": true,
    "last_attempt_time": 0,
    "last_success_time": 0
}
```

Dashboards

Dashboards are built-in or customized views of your ExtraHop metrics information. For more information, see Dashboards.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /dashboards	Retrieve all dashboards.
DELETE /dashboards/{id}	Delete a specific dashboard.
GET /dashboards/{id}	Retrieve a specific dashboard.
PATCH /dashboards/{id}	Update ownership of a specific dashboard.
GET /dashboards/{id}/reports	Retrieve scheduled reports that contain a specific dashboard. **Note:** This operation is only available from a console.
GET /dashboards/{id}/sharing	Retrieve the users and their sharing permissions for a specific dashboard.
PATCH /dashboards/{id}/sharing	Update the users and their sharing permissions for a specific dashboard.
PUT /dashboards/{id}/sharing	Replace the users and their sharing permissions for a specific dashboard.

Operation details

GET /dashboards

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "author": "string",
    "comment": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string",
    "owner": "string",
    "rights": [
        "string"
    ],
    "short_code": "string",
    "type": "string"
}
```

GET /dashboards/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the dashboard.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "author": "string",
    "comment": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string",
    "owner": "string",
    "rights": [
        "string"
    ],
    "short_code": "string",
    "type": "string"
}
```

DELETE /dashboards/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the dashboard.

PATCH /dashboards/{id}

Specify the following parameters.

body: ***Object***
The username of the dashboard owner.
id: ***Number***
The unique identifier for the dashboard.

GET /dashboards/{id}/sharing

Specify the following parameters.

id: ***Number***
The unique identifier for the dashboard.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "anyone": "string",
    "groups": {},
    "users": {}
}
```

PUT /dashboards/{id}/sharing

Specify the following parameters.

body: ***Object***
The users and their permission levels.
id: ***Number***
The unique identifier for the dashboard.

PATCH /dashboards/{id}/sharing

Specify the following parameters.

body: ***Object***
The users and their permission levels.

id: ***Number***
The unique identifier for the dashboard.

GET /dashboards/{id}/reports

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

Device

Devices are objects on your network that have been identified and classified by your ExtraHop system. For more information, see Devices.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /devices	Retrieve all devices that were active within a specific time period. For more information, see Extract the device list through the REST API. **Note:** A device is considered inactive after five minutes of not sending or receiving packets. However, if a device resumes sending or receiving packets after a period of inactivity shorter than five days, the device is considered to have been active continuously, including during the period of inactivity.
POST /devices/search	Retrieve all devices that match specific criteria. For more information, see Search for a device through the REST API. **Note:** A device is considered inactive after five minutes of not sending or receiving packets. However, if a device resumes sending or receiving packets after a period of inactivity shorter than five days, the device is considered to have been active continuously, including during the period of inactivity.
GET /devices/{id}	Retrieve a specific device.
PATCH /devices/{id}	Update a specific device.
GET /devices/{id}/activity	Retrieve all activity for a device.
GET /devices/{id}/alerts	Retrieve all alerts that are assigned to a specific device.

Operation	Description
POST /devices/{id}/alerts	Assign and unassign a specific device to alerts.
DELETE /devices/{id}/alerts/{child-id}	Unassign an alert from a specific device.
POST /devices/{id}/alerts/{child-id}	Assign an alert to a specific device.
GET /devices/{id}/dashboards	Retrieve all dashboards related to a specific device.
GET /devices/{id}/devicegroups	Retrieve all device groups that are assigned to a specific device.
POST /devices/{id}/devicegroups	Assign and unassign a specific device to device groups.
DELETE /devices/{id}/devicegroups/{child-id}	Unassign a device group from a specific device.
POST /devices/{id}/devicegroups/{child-id}	Assign a device group to a specific device.
GET /devices/{id}/dnsnames	Retrieve all DNS names that are associated with a specific device.
GET /devices/{id}/ipaddrs	Retrieve all IP addresses that were associated with a specific device within a given time period.
GET /devices/{id}/software	Retrieve a list of software running on the specified device.
GET /devices/{id}/tags	Retrieve all tags that are assigned to a specific device.
POST /devices/{id}/tags	Assign and unassign a specific device to tags.
DELETE /devices/{id}/tags/{child-id}	Unassign a tag from a specific device.
POST /devices/{id}/tags/{child-id}	Assign a tag to a specific device.
GET /devices/{id}/triggers	Retrieve all triggers that are assigned to a specific device.
POST /devices/{id}/triggers	Assign and unassign a specific device to triggers.
DELETE /devices/{id}/triggers/{child-id}	Unassign a trigger from a specific device.
POST /devices/{id}/triggers/{child-id}	Assign a trigger to a specific device.

Operation details

GET /devices

Specify the following parameters.

active_from: ***Number***

(Optional) The beginning timestamp for the request. Return only devices active after this time. Time is expressed in milliseconds since the epoch. 0 indicates the time of the request. A negative value is evaluated relative to the current time. The default unit for a negative value is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

active_until: ***Number***

(Optional) The ending timestamp for the request. Return only device active before this time. Follows the same time value guidelines as the active_from parameter.

limit: ***Number***

(Optional) Limit the number of devices returned to the specified maximum number.

offset: ***Number***

(Optional) Skip the first n device results. This parameter is often combined with the limit parameter.

search_type: ***String***

Indicates the field to search.

The following values are valid:

- `any`
- `name`
- `discovery_id`
- `ip address`
- `mac address`
- `vendor`
- `type`
- `tag`
- `activity`
- `node`
- `vlan`
- `discover time`

value: ***String***

(Optional) Specifies the search criteria.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "activity": [],
    "analysis": "string",
    "analysis_level": 0,
    "auto_role": "string",
    "cdp_name": "string",
    "cloud_account": "string",
    "cloud_instance_description": "string",
    "cloud_instance_id": "string",
    "cloud_instance_name": "string",
    "cloud_instance_type": "string",
    "critical": true,
    "custom_criticality": "string",
    "custom_make": "string",
    "custom_model": "string",
    "custom_name": "string",
    "custom_type": "string",
    "default_name": "string",
    "description": "string",
    "device_class": "string",
    "dhcp_name": "string",
    "discover_time": 0,
    "discovery_id": "string",
    "display_name": "string",
    "dns_name": "string",
    "extrahop_id": "string",
    "id": 0,
    "ipaddr4": "string",
    "ipaddr6": "string",
    "is_l3": true,
    "last_seen_time": 0,
    "macaddr": "string",
```

```
    "mod_time": 0,
    "model": "string",
    "model_override": "string",
    "netbios_name": "string",
    "node_id": 0,
    "on_watchlist": true,
    "parent_id": 0,
    "role": "string",
    "subnet_id": "string",
    "user_mod_time": 0,
    "vendor": "string",
    "vlanid": 0,
    "vpc_id": "string"
}
```

POST /devices/search

Specify the following parameters.

body: ***Object***

The device criteria.

active_from: ***Number***

(Optional) The beginning timestamp for the request. Return only devices active after this time. Time is expressed in milliseconds since the epoch. 0 indicates the time of the request. A negative value is evaluated relative to the current time. The default unit for a negative value is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

active_until: ***Number***

(Optional) The ending timestamp for the request. Return only devices active before this time. Follows the same time value guidelines as the active_from parameter.

limit: ***Number***

(Optional) Limit the number of devices returned to the specified maximum number.

offset: ***Number***

(Optional) Skip the specified number of devices. This parameter is often combined with the limit parameter to paginate result sets.

filter: ***Object***

(Optional) Specify the filter criteria for search results.

field: ***String***

The name of the field to filter results on. The search compares the contents of the field parameter to the value of the operand parameter.

The following values are valid:

- `name`
- `discovery_id`
- `ipaddr`
- `macaddr`
- `vendor`
- `tag`
- `activity`
- `node`
- `vlan`
- `discover_time`
- `role`

- `dns_name`
- `dhcp_name`
- `netbios_name`
- `cdp_name`
- `custom_name`
- `software`
- `model`
- `is_critical`
- `instance_id`
- `instance_name`
- `instance_type`
- `cloud_account`
- `vpc_id`
- `subnet_id`
- `is_active`
- `analysis`

operator: ***String***

The compare method applied when matching the operand value against the field contents. All filter objects require an operator.

The following values are valid:

- `>`
- `<`
- `<=`
- `>=`
- `=`
- `!=`
- `startswith`
- `and`
- `or`
- `not`
- `exists`
- `not_exists`
- `~`
- `!~`

operand: ***String or Number or Object***

The value that the query attempts to match. The query compares the value of the operand to the contents of the field parameter and applies the compare method specified by the operator parameter. You can specify the operand as a string, integer, or object. For information about object values, see the REST API Guide.

rules: ***Array of Objects***

An array of one or more filter objects, which can be embedded recursively. Only "and", "or", and "not" operators are allowed for this parameter.

Specify the body parameter in the following JSON format.

```
{
    "active_from": 0,
    "active_until": 0,
    "filter": {
        "field": "string",
        "operator": "string",
```

```
        "operand": "string",
        "rules": []
    },
    "limit": 0,
    "offset": 0
}
```

GET /devices/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "activity": [],
    "analysis": "string",
    "analysis_level": 0,
    "auto_role": "string",
    "cdp_name": "string",
    "cloud_account": "string",
    "cloud_instance_description": "string",
    "cloud_instance_id": "string",
    "cloud_instance_name": "string",
    "cloud_instance_type": "string",
    "critical": true,
    "custom_criticality": "string",
    "custom_make": "string",
    "custom_model": "string",
    "custom_name": "string",
    "custom_type": "string",
    "default_name": "string",
    "description": "string",
    "device_class": "string",
    "dhcp_name": "string",
    "discover_time": 0,
    "discovery_id": "string",
    "display_name": "string",
    "dns_name": "string",
    "extrahop_id": "string",
    "id": 0,
    "ipaddr4": "string",
    "ipaddr6": "string",
    "is_l3": true,
    "last_seen_time": 0,
    "macaddr": "string",
    "mod_time": 0,
    "model": "string",
    "model_override": "string",
    "netbios_name": "string",
    "node_id": 0,
    "on_watchlist": true,
    "parent_id": 0,
    "role": "string",
    "subnet_id": "string",
    "user_mod_time": 0,
    "vendor": "string",
    "vlanid": 0,
    "vpc_id": "string"
```

```
}
```

PATCH /devices/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property value updates to the device.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

GET /devices/{id}/activity

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "device_id": 0,
    "from_time": 0,
    "id": 0,
    "mod_time": 0,
    "stat_name": "string",
    "until_time": 0
}
```

GET /devices/{id}/ipaddrs

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

from: ***Number***

(Optional) Retrieves IP addresses that were associated with the device after the specified date, expressed in milliseconds since the epoch.

until: ***Number***

(Optional) Retrieves IP addresses that were associated with the device before the specified date, expressed in milliseconds since the epoch.

GET /devices/{id}/dnsnames

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

from: ***Number***

(Optional) Retrieves DNS names that were associated with the device after the specified date, expressed in milliseconds since the epoch.

until: ***Number***

(Optional) Retrieves DNS names that were associated with the device before the specified date, expressed in milliseconds since the epoch.

GET /devices/{id}/triggers

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

direct_assignments_only: ***Boolean***

(Optional) Restrict results to only triggers that are directly assigned to the device.

POST /devices/{id}/triggers

Specify the following parameters.

body: ***Object***

A list of unique identifiers for triggers that are assigned and unassigned to the device.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

POST /devices/{id}/triggers/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the trigger.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

DELETE /devices/{id}/triggers/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the trigger.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

GET /devices/{id}/dashboards

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

GET /devices/{id}/devicegroups

Specify the following parameters.

id: ***Number***

The unique identifier for the device.

active_from: ***Number***

(Optional) The beginning timestamp for the request. Return only dynamic device groups that the device belonged to after this time. Time is expressed in milliseconds since the epoch. 0 indicates the time of the request. A negative value is evaluated relative to the current time. The default unit for a negative value is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

active_until: ***Number***

(Optional) The ending timestamp for the request. Return only dynamic device groups that the device belonged to before this time. Follows the same time value guidelines as the active_from parameter.

POST /devices/{id}/devicegroups

Specify the following parameters.

body: ***Object***

The list of unique identifiers for device groups that are assigned and unassigned to the device.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

POST /devices/{id}/devicegroups/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device group.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

DELETE /devices/{id}/devicegroups/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device group.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

GET /devices/{id}/tags

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

POST /devices/{id}/tags

Specify the following parameters.

body: ***Object***

A list of unique identifiers for tags that are assigned and unassigned to the device.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

POST /devices/{id}/tags/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the tag.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

DELETE /devices/{id}/tags/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the tag.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

GET /devices/{id}/alerts

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

direct_assignments_only: ***Boolean***

(Optional) Restrict results to only alerts that are directly assigned to the device.

POST /devices/{id}/alerts

Specify the following parameters.

body: ***Object***

The list of unique identifiers for alerts that are assigned and unassigned to the device.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

POST /devices/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the alert.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

DELETE /devices/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the alert.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

GET /devices/{id}/software

Specify the following parameters.

id: ***Number***

The unique identifier for the device, which is displayed as the API ID on the device page in the ExtraHop system.

from: ***Number***

(Optional) Returns software that was observed on the device after the specified date, expressed in milliseconds since the epoch.

until: ***Number***

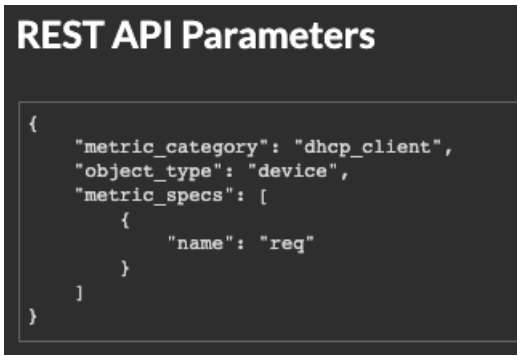
(Optional) Returns software that was observed on the device before the specified date, expressed in milliseconds since the epoch.

Operand values for device search

The POST /devices/search operation enables you to search for devices by criteria specified in filter objects. Each object should contain a unique value for the `operand` field that is valid for the specified `field` value.

`activity`

To search by metric activity, specify the `field` value as `activity` and the `operand` value as a `metric_category`. You can find `metric_category` values in the REST API Parameters section of the Metric Catalog.



The following example returns results for devices that match all metric activity classified for a DHCP client, such as the number of DHCP requests sent.

```
{
    "filter":  {
        "field": "activity",
        "operand": "dhcp_client",
        "operator": "="
    }
}
```

Tip: Programmatically retrieve a list of all metric activity for a device through the `GET /devices/{id}/activity` operation. The `stat_name` value matches the `metric_category` value in the `metric_catalog`, after the final dot.

In the following example response, the `stat_name` value is `extrahop.device.dhcp_client`. Remove the text before the final dot to identify the `metric_catalog` value of `dhcp_client`.

```
{
    "id": 198606,
    "from_time": 1581537120000,
    "until_time": 1581542520000,
```

```
    "mod_time": 1581542533963,
    "device_id": 30096,
    "stat_name": "extrahop.device.dhcp_client"
}
```

analysis

To search by device analysis level, specify the `field` value as `analysis` and the `operand` value as one of the following strings:

standard
Devices in Standard Analysis.

advanced
Devices in Advanced Analysis.

discovery
Devices in Discovery Mode.

l2_exempt
Devices in L2 Parent Analysis.

flow_log
Devices in Flow Analysis.

`discover_time`

To search by a time range, specify the `field` value as `discover_time` and an `operand` value with `from` and `until` parameters, where the values are dates, expressed in milliseconds since the epoch.

The following example returns results for all device activity that occurred between 1:00 PM to 3:00 PM on August 21, 2019.

```
{
    "filter":  {
        "field": "discover_time",
        "operand": {
            "from": "1566392400000",
            "until": "1566399600000"
        },
        "operator": "="
    }
}
```

`discovery_id`

To search by the unique ID for the device, specify the `field` value as `discovery_id` and the `operand` value as the discovery ID.

```
{
  "filter": {
    "field": "discovery_id",
    "operand": "c12vf90qpg290000",
    "operator": "="
  }
}
```

is_active

To search by devices that have activity in the last 30 minutes, specify the field value as `is_active` and the `operand` value as a boolean.

```
{
  "filter": {
    "field": "is_active",
    "operand": true,
    "operator": "="
  }
}
```

`ipaddr`

To search by IP address, specify the `field` value as `ipaddr` and the `operand` value as an IP address or CIDR block.

```
{
  "filter": {
    "field": "ipaddr",
    "operand": "192.168.12.0/28",
    "operator": "="
  }
}
```

`node`

To search by the unique ID of a sensor, specify the `field` value as `node` and the `operand` value as the sensor UUID.

```
{
  "filter": {
    "field": "node",
    "operand": "qqvsplfa-zxsk-32l0-19g1-076vfr42pw31",
    "operator": "="
  }
}
```

`macaddr`

To search by the MAC address of a device, specify the field value as `macaddr` and the operand value as the device MAC address. The following example returns results for devices with a MAC address of `C1:1C:N2:0Q:PJ:10` or `C1:1C:N2:0Q:PJ:11`.

```
{
  "filter": {
    "operator": "or",
    "rules": [
      {
        "field": "macaddr",
        "operand": "C1:1C:N2:0Q:PJ:10",
        "operator": "="
      },
      {
        "field": "macaddr",
        "operand": "C1:1C:N2:0Q:PJ:11",
        "operator": "="
      }
    ]
```

```
  }
}
```

`name`

To search by the device display name, specify the `field` value as name and the `operand` value as the device name or as a regex string.

```
{
  "filter": {
    "field": "name",
    "operand": "VMware B2CEB6",
    "operator": "="
  }
}
```

`role`

To search by the device role, specify the `field` value as `role` and the `operand` value as the device role.

```
{
  "filter": {
    "field": "role",
    "operand": "voip_phone",
    "operator": "="
  }
}
```

`software`

To search by the software running on the device, specify the `field` value as `software` and the `operand` value as the ID associated with that software on the ExtraHop system or as a regex string.

```
{
  "filter": {
    "field": "software",
    "operand": "windows_10",
    "operator": "="
  }
}
```

Tip: Programmatically retrieve a list of all software IDs associated with a device through the `GET /devices/{id}/software` operation.

In the following example response, the `id` value for the software is `windows_10`.

```
[
  {
    "software_type": "OS",
    "name": "Windows",
    "version": "10",
    "description": null,
    "id": "windows_10"
  }
]
```

`tag`

To search by a device tag, specify the `field` value as `tag` and the `operand` value as the tag name or as a regex string.

```
{
  "filter": {
    "field": "tag",
    "operand": "Custom Tag",
    "operator": "="
  }
}
```

Tip: Programmatically retrieve a list of all device tags through the `GET /devices/{id}/tags` operation.

In the following example response, the `name` value for the tag is `Custom Tag`.

```
[
  {
    "mod_time": 1521577040934,
    "id": 19,
    "name": "Custom Tag"
  }
]
```

vlan

To search by the ID of a VLAN, specify the `field` value as `vlan` and the `operand` value as the ID of the VLAN.

```
{
  "filter": {
    "field": "vlan",
    "operand": "0",
    "operator": "="
  }
}
```

Search with regular expressions (regex)

For certain `field` values, the string can be in regex syntax. Specify the `operand` value as an object that has a `value` parameter with the regex syntax you want to match and an `is_regex` parameter that is set to `true`. The following example returns results for all DNS names that end with `com`.

```
{
    "filter":  {
        "field": "dns_name",
        "operand": {
            "value": ".*?com",
            "is_regex": true
        },
        "operator": "="
    }
}
```

An `operand` field with regex syntax is valid for the following `field` values:

- cdp_name
- custom_name

- dns_name
- dhcp_name
- model
- name
- netbios_name
- software
- tag
- vendor

Supported time units

For most parameters, the default unit for time measurement is milliseconds. However, the following parameters return or accept alternative time units such as minutes and hours:

- Device
  - active_from
  - active_until
- Device group
  - active_from
  - active_until
- Metrics
  - from
  - until
- Record Log
  - from
  - until
  - context_ttl

The following table displays supported time units:

Time unit	Unit suffix
Year	`y`
Month	`M`
Week	`w`
Day	`d`
Hour	`h`
Minute	`m`
Second	`s`
Millisecond	`ms`

To specify a time unit other than milliseconds for a parameter, append the unit suffix to the value. For example, to request devices active in the last 30 minutes, specify the following parameter value:

```
GET /api/v1/devices?active_from=-30m
```

The following example specifies a search for HTTP records created between 1 and 2 hours ago:

```
{
    "from": "-2h",
```

```
    "until": "-1h",
    "types": ["~http"]
}
```

Device group

Device groups can be either static or dynamic.

A static device group is user-defined; you create a device group and then manually identify and assign each device to that group. A dynamic device group is defined and automatically managed by a set of configured rules.

For example, you can create a device group and then set a rule to classify all devices within a certain IP address range to be added to that group automatically. For more information, see Device Groups.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /devicegroups	Retrieve all device groups that were active within a specific time period.
POST /devicegroups	Create a new device group.
DELETE /devicegroups/{id}	Delete a device group.
GET /devicegroups/{id}	Retrieve a specific device group.
PATCH /devicegroups/{id}	Update a specific device group.
GET /devicegroups/{id}/alerts	Retrieve all alerts that are assigned to a specific device group.
POST /devicegroups/{id}/alerts	Assign and unassign a specific device group to alerts.
DELETE /devicegroups/{id}/alerts/{child-id}	Unassign an alert from a specific device group.
POST /devicegroups/{id}/alerts/{child-id}	Assign an alert to a specific device group.
GET /devicegroups/{id}/dashboards	Retrieve all dashboards related to a specific device group.
GET /devicegroups/{id}/devices	Retrieve all devices in the device group that are active within a specific time window. **Note:** A device is considered inactive after five minutes of not sending or receiving packets. However, if a device resumes sending or receiving packets after a period of inactivity shorter than five days, the device is considered to have been active continuously, including during the period of inactivity.
POST /devicegroups/{id}/devices	Assign and unassign a devices to a specific static device group.
DELETE /devicegroups/{id}/devices/{child-id}	Unassign a device from a specific static device group.
POST /devicegroups/{id}/devices/{child-id}	Assign a device to a specific static device group.

Operation	Description
GET /devicegroups/{id}/triggers	Retrieve all triggers that are assigned to a specific device group.
POST /devicegroups/{id}/triggers	Assign and unassign a specific device group to triggers.
DELETE /devicegroups/{id}/triggers/{child-id}	Unassign a trigger from a specific device group.
POST /devicegroups/{id}/triggers/{child-id}	Assign a trigger to a specific device group.

Operation details

GET /devicegroups

Specify the following parameters.

since: ***Number***

(Optional) Only return device groups that were modified after this time, expressed in milliseconds since the epoch.

all: ***Boolean***

(Optional) Deprecated. Replaced by the type parameter.

name: ***String***

(Optional) The Regex search value to filter the device groups by name.

type: ***String***

(Optional) Only return device groups of the specified type.

The following values are valid:

- `user_created`
- `built_in`
- `all`

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "built_in": true,
    "description": "string",
    "dynamic": true,
    "field": "string",
    "filter": {},
    "id": 0,
    "include_custom_devices": true,
    "mod_time": 0,
    "name": "string",
    "value": "string"
}
```

GET /devicegroups/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the device group.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
```

```
    "built_in": true,
    "description": "string",
    "dynamic": true,
    "field": "string",
    "filter": {},
    "id": 0,
    "include_custom_devices": true,
    "mod_time": 0,
    "name": "string",
    "value": "string"
}
```

POST /devicegroups

Specify the following parameters.

body: ***Object***

Apply the specified property values to the new device group.

description: ***String***

An optional description of the device group.

name: ***String***

The friendly name for the device group.

include_custom_devices: ***Boolean***

(Optional) Deprecated. Replaced by the filter parameter.

dynamic: ***Boolean***

(Optional) Indicates whether the device group is dynamic.

field: ***String***

Deprecated. Replaced by the filter parameter.

The following values are valid:

- `any`
- `name`
- `ip address`
- `mac address`
- `vendor`
- `type`
- `tag`
- `vlan`
- `activity`
- `node`
- `discover time`

value: ***Object***

(Optional) Deprecated. Replaced by the filter parameter.

filter: ***Object***

(Optional) Specify the filter criteria for search results.

field: ***String***

The name of the field to filter results on. The search compares the contents of the field parameter to the value of the operand parameter.

The following values are valid:

- `name`
- `ipaddr`

- macaddr
- vendor
- tag
- activity
- node
- vlan
- discover_time
- role
- dns_name
- dhcp_name
- netbios_name
- cdp_name
- custom_name
- software
- model
- is_critical
- instance_id
- instance_name
- instance_type
- cloud_account
- vpc_id
- subnet_id
- is_active

operator: ***String***

The compare method applied when matching the operand value against the field contents. All filter objects require an operator.

The following values are valid:

- >
- <
- <=
- >=
- =
- !=
- startswith
- and
- or
- not
- exists
- not_exists
- ~
- !~

operand: ***String or Number or Object***

The value that the query attempts to match. The query compares the value of the operand to the contents of the field parameter and applies the compare method specified by the operator parameter. You can specify the operand as a string, integer, or object. For information about object values, see the REST API Guide.

rules: ***Array of Objects***

An array of one or more filter objects, which can be embedded recursively. Only "and", "or", and "not" operators are allowed for this parameter.

Specify the body parameter in the following JSON format.

```
{
    "description": "string",
    "dynamic": true,
    "field": "string",
    "filter": {
        "field": "string",
        "operator": "string",
        "operand": "string",
        "rules": []
    },
    "include_custom_devices": true,
    "name": "string",
    "value": "string"
}
```

DELETE /devicegroups/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the device group.

PATCH /devicegroups/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property value updates to a specific device group.

description: ***String***

An optional description of the device group.

name: ***String***

The friendly name for the device group.

include_custom_devices: ***Boolean***

(Optional) Deprecated. Replaced by the filter parameter.

field: ***String***

Deprecated. Replaced by the filter parameter.

The following values are valid:

- `any`
- `name`
- `ip address`
- `mac address`
- `vendor`
- `type`
- `tag`
- `vlan`
- `activity`
- `node`
- `discover time`

value: ***Object***

(Optional) Deprecated. Replaced by the filter parameter.

filter: ***Object***

(Optional) Specify the filter criteria for search results.

Specify the body parameter in the following JSON format.

```
{
    "description": "string",
    "field": "string",
    "filter": {},
    "include_custom_devices": true,
    "name": "string",
    "value": "string"
}
```

id: ***Number***

The unique identifier for the device group.

GET /devicegroups/{id}/alerts

Specify the following parameters.

id: ***Number***

The unique identifier for the device group.

direct_assignments_only: ***Boolean***

(Optional) Restrict results to only alerts that are directly assigned to the device group.

POST /devicegroups/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the alert.

id: ***Number***

The unique identifier for the device group.

DELETE /devicegroups/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the alert.

id: ***Number***

The unique identifier for the device group.

POST /devicegroups/{id}/alerts

Specify the following parameters.

body: ***Object***

The list of unique identifiers for alerts that is assigned and unassigned to the device group.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***
The unique identifier for the device group.

GET /devicegroups/{id}/triggers

Specify the following parameters.

id: ***Number***
The unique identifier for the device group.

direct_assignments_only: ***Boolean***
(Optional) Restrict results to only triggers that are directly assigned to the device group.

POST /devicegroups/{id}/triggers/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the trigger.

id: ***Number***
The unique identifier for the device group.

DELETE /devicegroups/{id}/triggers/{child-id}

Specify the following parameters.

child-id: ***Number***
The unique identifier for the trigger.

id: ***Number***
The unique identifier for the device group.

POST /devicegroups/{id}/triggers

Specify the following parameters.

body: ***Object***
The list of unique identifiers for triggers that is assigned and unassigned to the device group.

assign: ***Array of Numbers***
IDs of resources to assign

unassign: ***Array of Numbers***
IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the device group.

POST /devicegroups/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for a device.

id: ***Number***

The unique identifier for the device group.

DELETE /devicegroups/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for a device.

id: ***Number***

The unique identifier for the device group.

POST /devicegroups/{id}/devices

Specify the following parameters.

body: ***Object***

The list of unique identifiers for devices that is assigned and unassigned to the device group.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the device group.

GET /devicegroups/{id}/devices

Specify the following parameters.

id: ***Number***

The unique identifier for the device group.

active_from: ***Number***

(Optional) The beginning timestamp for the request. Return only devices active after this time. Time is expressed in milliseconds since the epoch. 0 indicates the time of the request. A negative value is evaluated relative to the current time. The default unit for a negative value is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

active_until: ***Number***

(Optional) The ending timestamp for the request. Return only device active before this time. Follows the same time value guidelines as the active_from parameter.

limit: ***Number***

(Optional) Limit the number of devices returned.

offset: ***Number***

(Optional) Skip the first n device results. This parameter is often combined with the limit parameter.

GET /devicegroups/{id}/dashboards

Specify the following parameters.

id: ***Number***

The unique identifier for the device group.

Supported time units

For most parameters, the default unit for time measurement is milliseconds. However, the following parameters return or accept alternative time units such as minutes and hours:

- Device
  - active_from
  - active_until
- Device group
  - active_from
  - active_until
- Metrics
  - from
  - until
- Record Log
  - from
  - until
  - context_ttl

The following table displays supported time units:

Time unit	Unit suffix
Year	`y`
Month	`M`
Week	`w`
Day	`d`
Hour	`h`
Minute	`m`
Second	`s`
Millisecond	`ms`

To specify a time unit other than milliseconds for a parameter, append the unit suffix to the value. For example, to request devices active in the last 30 minutes, specify the following parameter value:

```
GET /api/v1/devices?active_from=-30m
```

The following example specifies a search for HTTP records created between 1 and 2 hours ago:

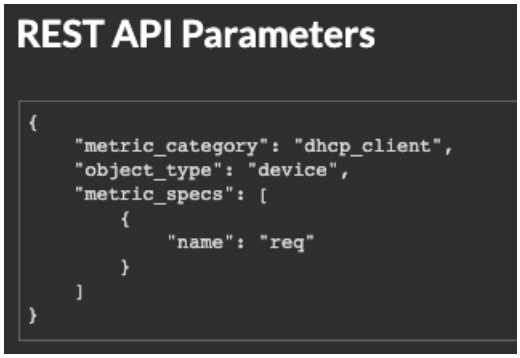
```
{
    "from": "-2h",
    "until": "-1h",
    "types": ["~http"]
}
```

Operand values for device groups

The POST /devicegroups operation enables you to create device groups according to criteria specified in filter objects. Each object should contain a unique value for the `operand` field that is valid for the specified `field` value.

`activity`

To select devices by metric activity, specify the `field` value as `activity` and the `operand` value as a `metric_category`. You can find `metric_category` values in the REST API Parameters section of the Metric Catalog.



The following example selects devices with metric activity classified for a DHCP client, such as the number of DHCP requests sent.

```
{
    "filter":  {
        "field": "activity",
        "operand": "dhcp_client",
        "operator": "="
    }
}
```

Tip: Programmatically retrieve a list of all metric activity for a device through the `GET /devices/{id}/activity` operation. The `stat_name` value matches the `metric_category` value in the `metric_catalog`, after the final dot.

In the following example response, the `stat_name` value is `extrahop.device.dhcp_client`. Remove the text before the final dot to identify the `metric_catalog` value of `dhcp_client`.

```
{
    "id": 198606,
    "from_time": 1581537120000,
    "until_time": 1581542520000,
```

```
    "mod_time": 1581542533963,
    "device_id": 30096,
    "stat_name": "extrahop.device.dhcp_client"
}
```

`discover_time`

To select devices by a time range, specify the `field` value as `discover_time` and an `operand` value with `from` and `until` parameters, where the values are dates, expressed in milliseconds since the epoch.

The following example selects devices with activity that occurred between 1:00 PM to 3:00 PM on August 21, 2019.

```
{
    "filter":  {
        "field": "discover_time",
        "operand": {
            "from": "1566392400000",
            "until": "1566399600000"
        },
        "operator": "="
    }
}
```

`discovery_id`

To select devices by unique device ID, specify the `field` value as `discovery_id` and the `operand` value as the discovery ID.

```
{
  "filter": {
    "field": "discovery_id",
    "operand": "c12vf90qpg290000",
    "operator": "="
  }
}
```

`ipaddr`

To select devices by IP address, specify the `field` value as `ipaddr` and the `operand` value as an IP address or CIDR block.

```
{
  "filter": {
    "field": "ipaddr",
    "operand": "192.168.12.0/28",
    "operator": "="
  }
}
```

`node`

To select devices by the unique ID of a sensor, specify the `field` value as `node` and the `operand` value as the appliance UUID.

```
{
  "filter": {
    "field": "node",
    "operand": "qqvsplfa-zxsk-32l0-19g1-076vfr42pw31",
```

```
    "operator": "="
  }
}
```

`macaddr`

To select devices by MAC address, specify the field value as `macaddr` and the operand value as the device MAC address. The following example returns results for devices with a MAC address of `C1:1C:N2:0Q:PJ:10` or `C1:1C:N2:0Q:PJ:11`.

```
{
  "filter": {
    "operator": "or",
    "rules": [
      {
        "field": "macaddr",
        "operand": "C1:1C:N2:0Q:PJ:10",
        "operator": "="
      },
      {
        "field": "macaddr",
        "operand": "C1:1C:N2:0Q:PJ:11",
        "operator": "="
      }
    ]
  }
}
```

`name`

To select devices by display name, specify the `field` value as name and the `operand` value as the device name or as a regex string.

```
{
  "filter": {
    "field": "name",
    "operand": "VMware B2CEB6",
    "operator": "="
  }
}
```

`role`

To select devices by role, specify the `field` value as `role` and the `operand` value as the device role.

```
{
  "filter": {
    "field": "role",
    "operand": "voip_phone",
    "operator": "="
  }
}
```

`software`

To select devices by the software running on the device, specify the `field` value as `software` and the `operand` value as the ID associated with that software on the ExtraHop system or as a regex string.

```
{
```

```
  "filter": {
    "field": "software",
    "operand": "windows_10",
    "operator": "="
  }
}
```

Tip: Programmatically retrieve a list of all software IDs associated with a device through the `GET /devices/{id}/software` operation.

In the following example response, the `id` value for the software is `windows_10`.

```
[
  {
    "software_type": "OS",
    "name": "Windows",
    "version": "10",
    "description": null,
    "id": "windows_10"
  }
]
```

`tag`

To select devices by tag, specify the `field` value as `tag` and the `operand` value as the tag name or as a regex string.

```
{
  "filter": {
    "field": "tag",
    "operand": "Custom Tag",
    "operator": "="
  }
}
```

Tip: Programmatically retrieve a list of all device tags through the `GET /devices/{id}/tags` operation.

In the following example response, the `name` value for the tag is `Custom Tag`.

```
[
  {
    "mod_time": 1521577040934,
    "id": 19,
    "name": "Custom Tag"
  }
]
```

vlan

To select devices by the ID of a VLAN, specify the `field` value as `vlan` and the `operand` value as the ID of the VLAN.

```
{
  "filter": {
    "field": "vlan",
    "operand": "0",
    "operator": "="
  }
}
```

Search with regular expressions (regex)

For certain `field` values, the string can be in regex syntax. Specify the `operand` value as an object that has a `value` parameter with the regex syntax you want to match and an `is_regex` parameter that is set to `true`. The following example selects devices with DNS names that end with `com`.

```
{
    "filter":  {
        "field": "dns_name",
        "operand": {
            "value": ".*?com",
            "is_regex": true
        },
        "operator": "="
    }
}
```

An `operand` field with regex syntax is valid for the following `field` values:

- cdp_name
- custom_name
- dns_name
- dhcp_name
- model
- name
- netbios_name
- software
- tag
- vendor

Specify multiple criteria

You can specify multiple criteria with the `rules` field. The following example returns results for devices with an IP address of `192.168.12.0` or `192.168.12.1`.

```
{
  "filter": {
    "operator": "or",
    "rules": [
      {
        "field": "ipaddr",
        "operand": "192.168.12.0",
        "operator": "="
      },
      {
        "field": "ipaddr",
        "operand": "192.168.12.1",
        "operator": "="
      }
    ]
  }
}
```

Note: You cannot specify more than 1000 rules for a device group.

Detections

The Detections class enables you to retrieve detections that have been identified by the ExtraHop system.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /detections	Retrieve all detections.
GET /detections/formats	Retrieve all detection types.
POST /detections/formats	Create a new custom detection type.
DELETE /detections/formats/{id}	Delete a specific custom detection type.
PATCH /detections/formats/{id}	Update a specific custom detection type.
GET /detections/rules/hiding	Retrieve all tuning rules.
POST /detections/rules/hiding	Create a tuning rule.
DELETE /detections/rules/hiding/{id}	Delete a tuning rule.
PATCH /detections/rules/hiding/{id}	Update a tuning rule.
POST /detections/search	Retrieve detections that match the specified search criteria.
PATCH /detections/tickets	Update a ticket associated with detections.
GET /detections/{id}	Retrieve a specific detection.
PATCH /detections/{id}	Update a detection.
DELETE /detections/{id}/notes	Delete the notes for a given detection.
GET /detections/{id}/notes	Retrieve the notes for a given detection.
PUT /detections/{id}/notes	Create or replace notes for a given detection.

Operation details

GET /detections/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the detection.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "appliance_id": 0,
    "assignee": "string",
    "categories": [
        "string"
    ],
    "description": "string",
    "end_time": 0,
    "id": 0,
    "is_user_created": true,
    "mitre_tactics": [],
    "mitre_techniques": [],
    "participants": [],
    "properties": {},
    "resolution": "string",
    "risk_score": 0,
    "start_time": 0,
```

```
    "status": "string",
    "ticket_id": "string",
    "ticket_url": "string",
    "title": "string",
    "type": "string",
    "update_time": 0
}
```

GET /detections

Specify the following parameters.

limit: ***Number***

(Optional) Limit the number of detections returned to the specified maximum number. A random selection of detections is returned.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "appliance_id": 0,
    "assignee": "string",
    "categories": [
        "string"
    ],
    "description": "string",
    "end_time": 0,
    "id": 0,
    "is_user_created": true,
    "mitre_tactics": [],
    "mitre_techniques": [],
    "participants": [],
    "properties": {},
    "resolution": "string",
    "risk_score": 0,
    "start_time": 0,
    "status": "string",
    "ticket_id": "string",
    "ticket_url": "string",
    "title": "string",
    "type": "string",
    "update_time": 0
}
```

POST /detections/search

Specify the following parameters.

body: ***Object***

The detection search parameters.

filter: ***Object***

Detection-specific filters.

category: ***String***

Return detections from the specified category.

assignee: ***Array of Strings***

Returns detections assigned to the specified user. Specify ".none" to search for unassigned detections or specify ".me" to search for detections assigned to the authenticated user.

ticket_id: ***Array of Strings***

Returns detections that are associated with the specified tickets. Specify ".none" to search for detections that are not associated with tickets.

status: ***Array of Strings***

Returns detections for tickets with the specified status. Specify ".none" to search for detections without a ticket status.

The following values are valid:

- `new`
- `in_progress`
- `closed`
- `acknowledged`

resolution: ***Array of Strings***

Returns detections for tickets with the specified resolution. Specify ".none" to search for detections without resolutions.

The following values are valid:

- `action_taken`
- `no_action_taken`

types: ***Array of Strings***

Returns detections with the specified types.

risk_score_min: ***Number***

Returns detections with risk scores greater than or equal to the specified value.

from: ***Number***

Returns detections that occurred after the specified date, expressed in milliseconds since the epoch. Detections that started before the specified date are returned if the detection was ongoing at that time.

limit: ***Number***

Returns no more than the specified number of detections.

offset: ***Number***

The number of detections to skip for pagination.

sort: ***Array of Objects***

Sorts returned detections by the specified fields. By default, detections are sorted by most recent update time and then ID in ascending order.

direction: ***String***

The order in which returned detections are sorted.

The following values are valid:

- `asc`
- `desc`

field: ***String***

The field to sort detections by.

until: ***Number***

Return detections that ended before the specified date, expressed in milliseconds since the epoch.

update_time: ***Number***

Return detections that were updated on or after the specified date, expressed in milliseconds since the epoch.

Specify the body parameter in the following JSON format.

```
{
    "filter": {
        "category": "string",
        "assignee": [],
        "ticket_id": [],
        "status": [],
        "resolution": [],
        "types": [],
        "risk_score_min": 0
    },
    "from": 0,
    "limit": 0,
    "offset": 0,
    "sort": {
        "direction": "string",
        "field": "string"
    },
    "until": 0,
    "update_time": 0
}
```

PATCH /detections/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the detection.

body: ***Object***

The detection parameters to update.

ticket_id: ***String***

The ID of the ticket associated with the detection.

assignee: ***String***

The assignee of the detection or the ticket associated with the detection.

status: ***String***

The status of the detection or the ticket associated with the detection.

The following values are valid:

- `new`
- `in_progress`
- `closed`
- `acknowledged`

resolution: ***String***

The resolution of the detection or the ticket associated with the detection.

The following values are valid:

- `action_taken`
- `no_action_taken`

participants: ***Array of Objects***

A list of devices and applications associated with the detection. You can modify specific fields for a participant, but you cannot add new participants to a detection.

id: ***Number***

The ID of the participant associated with the detection.

usernames: ***Array of Strings***

The usernames associated with the participant through the REST API.

origins: ***Array of Strings***

The origin IP addresses associated with the participant through the REST API.

Specify the body parameter in the following JSON format.

```
{
    "assignee": "string",
    "participants": {
        "id": 0,
        "usernames": [],
        "origins": []
    },
    "resolution": "string",
    "status": "string",
    "ticket_id": "string"
}
```

PATCH /detections/tickets

Specify the following parameters.

body: ***Object***

The detection ticketing values to update.

ticket_id: ***String***

The ID of the ticket associated with the detection.

assignee: ***String***

The assignee of the ticket associated with the detection.

status: ***String***

The status of the ticket associated with the detection.

The following values are valid:

- `new`
- `in_progress`
- `closed`
- `acknowledged`

resolution: ***String***

The resolution of the ticket associated with the detection.

The following values are valid:

- `action_taken`
- `no_action_taken`

Specify the body parameter in the following JSON format.

```
{
    "assignee": "string",
    "resolution": "string",
    "status": "string",
    "ticket_id": "string"
}
```

GET /detections/formats

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "author": "string",
    "display_name": "string",
    "is_user_created": true,
    "mitre_categories": [],
    "properties": {},
    "type": "string"
}
```

POST /detections/formats

Specify the following parameters.

body: ***Object***

The parameters of the detection format.

type: ***String***

A string identifier for the detection type. The string can only contain letters, numbers, and underscores. Although detection types are unique across built-in formats, and detection types are unique across custom formats, a built-in and custom format can share the same detection type.

display_name: ***String***

The display name of the detection type that appears on the Detections page in the ExtraHop system.

mitre_categories: ***Array of Strings***

(Optional) The IDs of the MITRE techniques associated with the detection.

author: ***String***

(Optional) The author of the detection format.

Specify the body parameter in the following JSON format.

```
{
    "author": "string",
    "display_name": "string",
    "mitre_categories": [],
    "type": "string"
}
```

DELETE /detections/formats/{id}

Specify the following parameters.

id: ***String***

The string identifier of the detection format.

PATCH /detections/formats/{id}

Specify the following parameters.

id: ***String***

The string identifier of the detection format.

body: ***Object***

The parameters of the detection format.

GET /detections/rules/hiding

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "author": "string",
    "create_time": 0,
    "description": "string",
    "detection_type": "string",
    "detections_hidden": 0,
    "enabled": true,
    "expiration": 0,
    "hide_past_detections": true,
    "id": 0,
    "offender": {},
    "properties": [],
    "victim": {}
}
```

POST /detections/rules/hiding

Specify the following parameters.

body: ***Object***

The tuning rule parameters.

offender: ***Object***

The offender that this tuning rule applies to. Specify a detection_hiding_participant object to apply the rule to a specific victim, or specify "Any" to apply the rule to any offender.

object_type: ***String***

The type of participant.

The following values are valid:

- `device`
- `device_group`
- `ipaddr`

object_id: ***Number***

The ID for the device or device group participant. This option is valid only if the object_type is "device" or "device_group."

object_value: ***Array or String***

The IP address or CIDR block of the participant. You can specify a single address or block in a string or multiple addresses or blocks in an array. This option is valid only if the object_type is "ipaddr."

victim: ***Object***

The victim that this tuning rule applies to. Specify a detection_hiding_participant object to apply the rule to a specific victim, or specify "Any" to apply the rule to any victim.

object_type: ***String***

The type of participant.

The following values are valid:

- `device`
- `device_group`
- `ipaddr`

object_id: ***Number***

The ID for the device or device group participant. This option is valid only if the object_type is "device" or "device_group."

object_value: ***Array or String***

The IP address or CIDR block of the participant. You can specify a single address or block in a string or multiple addresses or blocks in an array. This option is valid only if the object_type is "ipaddr."

expiration: ***Number***

The time that the tuning rule expires, expressed in milliseconds since the epoch or as null to indicate the rule does not expire.

description: ***String***

(Optional) The description of the tuning rule.

detection_type: ***String***

The detection_type that this tuning rule applies to. Specify "All" to apply the rule to all detection types.

properties: ***Array of Objects***

(Optional) The filter criteria for detection properties.

property: ***String***

The name of the property to filter.

operator: ***String***

The compare method applied when matching the operand value against the detection property value.

The following values are valid:

- `=`
- `!=`
- `~`
- `!~`
- `in`

operand: ***String or Number or Object***

The value that the filter attempts to match. The filter compares the value of the operand to the value of the detection property and applies the compare method specified by the operator parameter. You can specify the operand as a string, integer, or object. For more information, see the REST API Guide.

Specify the body parameter in the following JSON format.

```
{
    "description": "string",
    "detection_type": "string",
    "expiration": 0,
    "offender": {
        "object_type": "string",
        "object_id": 0,
        "object_value": "array"
    },
    "properties": {
        "property": "string",
        "operator": "string",
        "operand": "string"
```

```
    },
    "victim": {
        "object_type": "string",
        "object_id": 0,
        "object_value": "array"
    }
}
```

PATCH /detections/rules/hiding/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the tuning rule.

body: ***Object***

The tuning rule fields to update.

enabled: ***Boolean***

Indicates whether the tuning rule is enabled.

expiration: ***Number***

The time that the tuning rule expires, expressed in milliseconds since the epoch or as null to indicate the rule does not expire.

Specify the body parameter in the following JSON format.

```
{
    "enabled": true,
    "expiration": 0
}
```

DELETE /detections/rules/hiding/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the tuning rule.

GET /detections/{id}/notes

Specify the following parameters.

id: ***Number***

The unique identifier for the detection.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "author": "string",
    "note": "string",
    "update_time": 0
}
```

DELETE /detections/{id}/notes

Specify the following parameters.

id: ***Number***

The unique identifier for the detection.

PUT /detections/{id}/notes

Specify the following parameters.

id: ***Number***

The unique identifier for the detection.

body: ***Object***

The detection note parameters.

Operand values for detection property tuning rules

The `POST /detections/rules/hiding` operation enables you to create tuning rules that filter detections based on detection properties. You can specify filtering criteria for detection properties in objects. Each object should contain a unique value for the `operand` field that is valid for the specified `property` value.

Tip: You can retrieve valid property values through the `GET /detections/formats` operation. See the keys of the `properties` object in the response. In the following example, the `property` value is `s3_bucket`:

```
    "properties": {
      "s3_bucket": {
        "is_optional": true,
        "status": "active",
        "is_tunable": true,
        "data_type": "string"
      }
    }
```

The `is_tunable` field indicates whether you can create a tuning rule based on the property.

`registered_domain_name`

To hide rules by a registered domain name, specify the `property` value as `registered_domain_name` and the `operand` value as a domain name.

The following example rule hides DNS Tunnel detections for `example.com`.

```
{
    "detection_type": "dns_tunnel",
    "expiration": null,
    "offender": "Any",
    "victim": "Any",
    "properties": [
        {
            "operand": "example.com",
            "operator": "=",
            "property": "registered_domain_name"
        }
    ]
}
```

`uris`

To hide rules by a URI, specify the `property` value as `uris` and the `operand` value as a URI.

The following example rule hides SQL Injection (SQLi) Attack detections for `http://example.com/test`.

```
{
    "detection_type": "sqli_attack",
```

```
    "expiration": null,
    "offender": "Any",
    "victim": "Any",
    "properties": [
        {
            "operand": "http://example.com/test",
            "operator": "=",
            "property": "uris"
        }
      ]
}
```

`top_level_domain`

To hide rules by a top-level domain name, specify the `property` value as `top_level_domain` and the `operand` value as a top-level domain name.

The following example rule hides Suspicious Top-level Domain detections for the `org` top-level domain.

```
{
    "detection_type": "suspicious_tld",
    "expiration": null,
    "offender": "Any",
    "victim": "Any",
    "properties": [
        {
            "operand": "org",
            "operator": "=",
            "property": "top_level_domain"
        }
    ]
}
```

Search with regular expressions (regex)

For certain `property` values, the string can be in regex syntax. Specify the `operand` value as an object that has a `value` parameter with the regex syntax you want to match and an `is_regex` parameter that is set to `true`. The following rule filters DNS Tunnel detections with domain names that end with `example.com`.

```
{
    "detection_type": "dns_tunnel",
    "expiration": null,
    "offender": "Any",
    "victim": "Any",
    "properties": [
        {
            "operand": {
                "value": ".*?example.com",
                "is_regex": true
            },
            "operator": "=",
            "property": "registered_domain_name"
        }
    ]
}
```

Disable case sensitivity

By default, searches for string `property` values are case-sensitive. However, you can disable case sensitivity by specifying the operand value as an object that has a `case_sensitive` parameter that is set to `false`.

The following rule hides Hacking Tool Domain Access detections with the ArchStrike hacking tool.

```
{
    "detection_type": "hacking_tools",
    "expiration": null,
    "offender": "Any",
    "victim": "Any",
    "properties": [
        {
            "operand": {
                "value": "archstrike",
                "case_sensitive": false
            },
            "operator": "=",
            "property": "hacking_tool"
        }
    ]
}
```

Email group

You can add individual or group email addresses to an email group and assign them to a system alert. When that alert is triggered, the system sends an email to all of the addresses in the email group.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /emailgroups	Retrieve all email groups.
POST /emailgroups	Create a new email group.
DELETE /emailgroups/{id}	Delete a email group by a unique identifier.
GET /emailgroups/{id}	Retrieve a specific email group by a unique identifier.
PATCH /emailgroups/{id}	Apply updates to a specific email group.

Operation details

GET /emailgroups

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "email_addresses": [],
    "group_name": "string",
    "id": 0,
    "system_notifications": true
}
```

POST /emailgroups

Specify the following parameters.

body: ***Object***

Apply the specified property values to the new email group.

group_name: ***String***

The friendly name for the email group.

email_addresses: ***Array of Strings***

The list of email addresses in the email group.

system_notifications: ***Boolean***

Indicates whether that the group should receive system notifications.

Specify the body parameter in the following JSON format.

```
{
    "email_addresses": [],
    "group_name": "string",
    "system_notifications": true
}
```

GET /emailgroups/{id}

Specify the following parameters.

id: ***Number***

The unique identifier of the email group.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "email_addresses": [],
    "group_name": "string",
    "id": 0,
    "system_notifications": true
}
```

DELETE /emailgroups/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the email group.

PATCH /emailgroups/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property value updates to the email group.

id: ***Number***

The unique identifier for the email group.

Exclusion intervals

An exclusion interval can be created to set a time period to suppress an alert.

For example, if you do not want to be notified about alerts after hours or on the weekends, an exclusion interval can create a rule to suppress the alert during that time period. For more information, see Alerts.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /exclusionintervals	Retrieve all exclusion intervals.
POST /exclusionintervals	Create a new exclusion interval.
DELETE /exclusionintervals/{id}	Delete a specific exclusion interval.
GET /exclusionintervals/{id}	Retrieve a specific exclusion interval.
PATCH /exclusionintervals/{id}	Apply updates to a specific exclusion interval.

Operation details

GET /exclusionintervals

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "alert_apply_all": true,
    "author": "string",
    "description": "string",
    "end": 0,
    "id": 0,
    "interval_type": "string",
    "mod_time": 0,
    "name": "string",
    "start": 0,
    "trend_apply_all": true
}
```

POST /exclusionintervals

Specify the following parameters.

body: ***Object***

Set the specified property values on the new exclusion interval.

name: ***String***

The friendly name for the exclusion interval.

author: ***String***

(Optional) The name of the creator of the exclusion interval.

description: ***String***

(Optional) An optional description of the exclusion interval.

interval_type: ***String***

The time window when the exclusion interval was evaluated.

The following values are valid:

- `onetime`
- `weekly`
- `daily`

start: ***Number***

The start of the exclusion interval time range, expressed in seconds. This value is relative to the epoch for onetime exclusions, relative to midnight for daily exclusions, and relative to Monday at midnight for weekly exclusions.

end: ***Number***

The end of the exclusion interval time range, expressed in seconds. This value is relative to the epoch for onetime exclusions, relative to midnight for daily exclusions, and relative to Monday at midnight for weekly exclusions.

alert_apply_all: ***Boolean***

Indicates whether this exclusion interval should be applied to all alerts.

trend_apply_all: ***Boolean***

Indicates whether this exclusion interval should be applied to all trends.

Specify the body parameter in the following JSON format.

```
{
    "alert_apply_all": true,
    "author": "string",
    "description": "string",
    "end": 0,
    "interval_type": "string",
    "name": "string",
    "start": 0,
    "trend_apply_all": true
}
```

GET /exclusionintervals/{id}

Specify the following parameters.

id: ***Number***

The unique identifier of the exclusion interval.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "alert_apply_all": true,
    "author": "string",
    "description": "string",
    "end": 0,
    "id": 0,
    "interval_type": "string",
    "mod_time": 0,
    "name": "string",
    "start": 0,
    "trend_apply_all": true
}
```

DELETE /exclusionintervals/{id}

Specify the following parameters.

id: ***Number***

The unique identifier of the exclusion interval.

PATCH /exclusionintervals/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property value updates to the exclusion interval.

id: ***Number***

The unique identifier for the exclusion interval.

ExtraHop

This resource provides metadata about the ExtraHop system.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /extrahop	Retrieve metadata about the firmware running on the ExtraHop system.
GET /extrahop/cluster	Retrieve Explore cluster configuration settings.
PATCH /extrahop/cluster	Update Explore cluster configuration settings.
GET /extrahop/detections/access	Retrieve the detections access control settings.
PUT /extrahop/detections/access	Update detections access control settings.
GET /extrahop/edition	Retrieve the edition of the ExtraHop system.
POST /extrahop/firmware	Upload a new firmware image to the ExtraHop system. For more information, see Upgrade ExtraHop firmware through the REST API.
POST /extrahop/firmware/download/url	Download a new firmware image onto the ExtraHop system from a URL.
POST /extrahop/firmware/download/version	Download a new firmware image onto the ExtraHop system from ExtraHop Cloud Services.
POST /extrahop/firmware/latest/upgrade	Upgrade the ExtraHop system to the most recently uploaded firmware image.
GET /extrahop/firmware/next	Upgrade the ExtraHop system to the most recently uploaded firmware image.
GET /extrahop/firmware/previous	Retrieve information about a firmware version that you can roll back the ExtraHop system to.
POST /extrahop/firmware/previous/rollback	Roll back the ExtraHop system to the previous firmware version.
GET /extrahop/flowlogs/secret	Retrieve the flow log secret.
POST /extrahop/flowlogs/secret	Generate a new flow log secret.
GET /extrahop/idrac	Retrieve the iDRAC IP address of the ExtraHop system.
GET /extrahop/platform	Retrieve the platform name of the ExtraHop system.
GET /extrahop/processes	Retrieve a list of processes running on the ExtraHop system.
POST /extrahop/processes/{process}/restart	Restart a process running on the ExtraHop system.
GET /extrahop/services	Retrieve settings for all services.
PATCH /extrahop/services	Update the settings for services.

Operation	Description
POST /extrahop/restart	Restart the ExtraHop system.
POST /extrahop/shutdown	Shut down the ExtraHop system.
POST /extrahop/sslcert	Regenerate the SSL certificate on the ExtraHop system. For more information, see Create a trusted SSL certificate through the REST API
PUT /extrahop/sslcert	Replace the SSL certificate on the ExtraHop system.
POST /extrahop/sslcert/signingrequest	Create an SSL certificate signing request. For more information, see Create a trusted SSL certificate through the REST API.
GET /extrahop/ticketing	Retrieve the ticketing integration status.
PATCH /extrahop/ticketing	Enable or disable ticketing integration.
GET /extrahop/version	Retrieve the version of the firmware running on the ExtraHop system.

Operation details

GET /extrahop/version

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "version": "string"
}
```

GET /extrahop/platform

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "platform": "string"
}
```

GET /extrahop/edition

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "edition": "string"
}
```

GET /extrahop

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "display_host": "string",
    "external_hostname": "string",
    "hostname": "string",
    "mgmt_ipaddr": "string",
    "platform": "string",
    "version": "string"
}
```

GET /extrahop/idrac

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "ipaddr": "string"
}
```

POST /extrahop/sslcert

There are no parameters for this operation.

PUT /extrahop/sslcert

Specify the following parameters.

body: ***String***

The SSL certificate and optionally the private key. Enter as plain text, separated with a line break.

POST /extrahop/sslcert/signingrequest

Specify the following parameters.

body: ***Object***

Parameters for the SSL certificate signing request.

subject_alternative_names: ***Array of Objects***

A list of names that the certificate applies to, such as {"type": "dns", "name": "www.example.com"}.

type: ***String***

Type of Subject Alternative Name.

The following values are valid:

- `dns`
- `ip`

name: ***String***

Name of Subject Alternative Name.

subject: ***Object***

The subject of the SSL certificate. For a list of certificate subject fields, see below.

common_name: ***String***

The subject common name (CN).

country_code: ***String***

(Optional) The subject country (C).

state_or_province_name: ***String***

(Optional) The subject state or province (ST).

locality_name: ***String***

(Optional) The subject locality (L).

organization_name: ***String***

(Optional) The subject organization (O).

organizational_unit_name: ***String***

(Optional) The subject organizational unit (OU).

email_address: ***String***

(Optional) The subject e-mail address (emailAddress).

Specify the body parameter in the following JSON format.

```
{
    "subject": {
        "common_name": "string",
        "country_code": "string",
        "state_or_province_name": "string",
        "locality_name": "string",
        "organization_name": "string",
        "organizational_unit_name": "string",
        "email_address": "string"
    },
    "subject_alternative_names": {
        "type": "string",
        "name": "string"
    }
}
```

GET /extrahop/ticketing

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "enabled": true,
    "url_template": "string"
}
```

PATCH /extrahop/ticketing

Specify the following parameters.

body: ***Object***

Ticket tracking settings.

enabled: ***Boolean***

(Optional) Indicates how detection investigations are tracked. If set to true, investigations are tracked from an external ticketing system. If set to false, investigations are tracked within the appliance.

url_template: ***String***

(Optional) The URL template that links detections to external tickets. The template must include the $ticket_id variable. This field applies only if detection investigations are tracked from an external ticketing system.

Specify the body parameter in the following JSON format.

```
{
    "enabled": true,
    "url_template": "string"
}
```

PUT /extrahop/detections/access

Specify the following parameters.

body: *Object*

The detections access settings for the appliance.

enabled: *Boolean*

Indicates whether detections access settings are enabled. When enabled, administrators can restrict detections access for specified users.

Specify the body parameter in the following JSON format.

```
{
    "enabled": true
}
```

GET /extrahop/detections/access

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "enabled": true
}
```

POST /extrahop/firmware

Specify the following parameters.

firmware: *Filename*

The .tar file that contains the firmware image. Note: You cannot upload a firmware image through the REST API explorer. For more information about how to upload an image through cURL or a Python script, see Upgrade ExtraHop firmware through the REST API.

POST /extrahop/firmware/latest/upgrade

Specify the following parameters.

body: *Object*

(Optional) The installation options for upgrading the appliance.

restart_after: *Boolean*

(Optional) Indicates whether to restart the appliance after the upgrade is complete.

silent: *Boolean*

(Optional) Specifies whether to disable the ExtraHop Web UI during the upgrade process. If an upgrade fails, the appliance will automatically revert to the previous firmware version.

force: *Boolean*

(Optional) Specifies whether to skip compatibility verification. Skip verification only if ExtraHop Support has reviewed and approved the upgrade.

Specify the body parameter in the following JSON format.

```
{
    "force": true,
    "restart_after": true,
    "silent": true
}
```

POST /extrahop/firmware/download/url

Specify the following parameters.

body: ***Object***

The download options.

firmware_url: ***String***

The URL of the firmware to download. HTTPS, HTTP, and FTP schemes are supported.

upgrade: ***Boolean***

(Optional) Specifies whether to upgrade the appliance after the firmware download is complete.

force: ***Boolean***

(Optional) Specifies whether to skip compatibility verification. Skip verification only if ExtraHop Support has reviewed and approved the upgrade.

Specify the body parameter in the following JSON format.

```
{
    "firmware_url": "string",
    "force": true,
    "upgrade": true
}
```

POST /extrahop/restart

There are no parameters for this operation.

POST /extrahop/shutdown

There are no parameters for this operation.

GET /extrahop/services

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "admin": {
        "enabled": true
    },
    "keyreceiver": {
        "enabled": true
    },
    "snmp": {
        "enabled": true
    },
    "ssh": {
        "enabled": true
    },
```

```
    "webshell": {
        "enabled": true
    }
}
```

PATCH /extrahop/services

Specify the following parameters.

body: ***Object***

The settings for services.

webshell: ***Object***

(Optional) The settings of the Web Shell service, which provides access to the ExtraHop command-line interface (CLI).

enabled: ***Boolean***

Indicates whether the service is enabled.

admin: ***Object***

(Optional) The settings of the Management GUI service, which provides browser-based access to the appliance.

enabled: ***Boolean***

Indicates whether the service is enabled.

snmp: ***Object***

(Optional) The settings of the SNMP service, which enables your network device monitoring software to collect information from the ExtraHop System.

enabled: ***Boolean***

Indicates whether the service is enabled.

ssh: ***Object***

(Optional) The settings of the SSH service, which enables users to securely log in to the ExtraHop command-line interface (CLI).

enabled: ***Boolean***

Indicates whether the service is enabled.

keyreceiver: ***Object***

(Optional) The settings of the SSL Session Key Receiver, which enables the appliance to receive and decrypt session keys from the session key forwarder.

enabled: ***Boolean***

Indicates whether the service is enabled.

Specify the body parameter in the following JSON format.

```
{
    "admin": {
        "enabled": true
    },
    "keyreceiver": {
        "enabled": true
    },
    "snmp": {
        "enabled": true
    },
    "ssh": {
        "enabled": true
    },
    "webshell": {
        "enabled": true
```

```
    }
}
```

GET /extrahop/processes

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "can_restart": true,
    "cpu": 0.0,
    "cpu_time": 0,
    "mem_percent": 0.0,
    "mem_res": 0,
    "mem_virt": 0,
    "process": "string",
    "start_time": 0
}
```

POST /extrahop/processes/{process}/restart

Specify the following parameters.

process: ***String***

The name of the process.

The following values are valid:

- `exadmin`
- `exalerts`
- `examf`
- `exapi`
- `exbridge`
- `excap`
- `exconfig`
- `exflowlogs`
- `exsnmpq`
- `exnotify`
- `exportal`
- `exremote`
- `exsearch`
- `extrend`
- `exwebshell`
- `webserver`

GET /extrahop/cluster

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "ingest_enabled": true,
    "replication_policy": 0
}
```

PATCH /extrahop/cluster

Specify the following parameters.

body: ***Object***

The EXA cluster configuration settings.

ingest_enabled: ***Boolean***

(Optional) Indicates whether record ingest is enabled for the Explore cluster.

replication_policy: ***Number***

(Optional) The replication level that determines how many copies of each record are stored.

Specify the body parameter in the following JSON format.

```
{
    "ingest_enabled": true,
    "replication_policy": 0
}
```

GET /extrahop/firmware/previous

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "backup_time": 0,
    "version": "string"
}
```

POST /extrahop/firmware/previous/rollback

There are no parameters for this operation.

GET /extrahop/flowlogs/secret

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "secret": "string"
}
```

POST /extrahop/flowlogs/secret

There are no parameters for this operation.

GET /extrahop/firmware/next

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "current_release": true,
    "release": "string",
    "versions": []
}
```

POST /extrahop/firmware/download/version

Specify the following parameters.

body: ***Object***

(Optional) The download options.

version: ***String***

The version of the firmware to download.

upgrade: ***Boolean***

(Optional) Specifies whether to upgrade the appliance after the firmware download is complete.

Specify the body parameter in the following JSON format.

```
{
    "upgrade": true,
    "version": "string"
}
```

Jobs

You can monitor the progress of some administration jobs started through the REST API. If a REST request creates a job, the job ID is returned in the `location` header of the response. The following operations create jobs:

- `POST /extrahop/firmware/latest/upgrade`
- `POST /extrahop/sslcert`

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /jobs	Retrieve the status of all jobs.
GET /jobs/{id}	Retrieve the status of a specific job.

Operation details

GET /jobs/{id}

Specify the following parameters.

id: ***String***

The unique identifier for the job.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "details": "string",
    "id": "string",
    "remote_jobs": [],
    "status": "string",
    "step_description": "string",
    "step_number": 0,
    "total_steps": 0,
    "type": "string"
}
```

GET /jobs

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "details": "string",
    "id": "string",
    "remote_jobs": [],
    "status": "string",
    "step_description": "string",
    "step_number": 0,
    "total_steps": 0,
    "type": "string"
}
```

Job types

The `GET /jobs` operation returns the following values in the `type` field of the response.

extrahop_firmware_download

The ExtraHop system is downloading a new firmware image from either a URL or ExtraHop Cloud Services.

extrahop_firmware_upgrade

The ExtraHop system is upgrading to a new firmware version.

extrahop_firmware_download_upgrade

The ExtraHop system is downloading a firmware image and upgrading to a new firmware version. The image is retrieved from either a URL or ExtraHop Cloud Services.

Note: The `type` field is empty for some jobs.

License

This resource enables you to retrieve and set product keys or to retrieve and set a license.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /license	Retrieve the license applied to this ExtraHop system.
PUT /license	Apply and register a new license to the ExtraHop system.
GET /license/productkey	Retrieve the product key to this ExtraHop system.
PUT /license/productkey	Apply the specified product key to the ExtraHop system and register the license.

Operation details

PUT /license

Specify the following parameters.

body: ***String***

(Optional) The license text provided to you by ExtraHop Support, including the BEGIN and END lines.

GET /license

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "dossier": "string",
    "edition": "string",
    "expires_at": 0,
    "expires_in": 0,
    "modules": {},
    "options": {},
    "platform": "string",
    "product_key": "string",
    "serial": "string"
}
```

PUT /license/productkey

Specify the following parameters.

body: ***Object***

(Optional) Apply the specified product key to the appliance.

GET /license/productkey

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "product_key": "string"
}
```

Metrics

Metrics information is collected about every object identified by the ExtraHop system.

Note that metrics are retrieved through the POST method, which creates a query to collect the requested information through the API. For more information, see Extract metrics through the REST API.

The following table displays all of the operations you can perform on this resource:

Operation	Description
POST /metrics	Perform a metric query.
GET /metrics/next/{xid}	If a previous metric query requested activity group metrics from a console, the GET /metrics/next/{xid} operation retrieves metrics for the activity group on a connected sensor. Each time a request is sent to GET /metrics/next/{xid}, the operation returns metrics from a different sensor. After all metrics have been retrieved, the operation returns null.

Operation	Description
POST /metrics/total	Perform a metric query for total values.
POST /metrics/totalbyobject	Perform a metric query for total values that are grouped by object.

For example, if you want to see all HTTP responses that occurred on the network in the last 30 minutes, enter the following request schema into the `POST /metrics` operation:

```
{
  "cycle": "auto",
  "from": -1800000,
  "metric_category": "http",
  "metric_specs": [
    {
      "name": "rsp"
    }
  ],
  "object_ids": [
    0
  ],
  "object_type": "application",
  "until": 0
}
```

The response body returns a list of HTTP responses and the time of each event, similar to the following output:

```
{
  "stats": [
    {
      "oid": 0,
      "time": 1494539640000,
      "duration": 30000,
      "values": [
        354
      ]
    },
    {
      "oid": 0,
      "time": 1494539640000,
      "duration": 30000,
      "values": [
        354
      ]
    },

    {
      "oid": 0,
      "time": 1494539640000,
      "duration": 30000,
      "values": [
        354
      ]
    },
  ],
  "cycle": "30sec",
  "node_id": 0,
  "clock": 1494541440000,
  "from": 1494539640000,
  "until": 1494541440000
```

```
}
```

Enter the same request schema into the `POST /metrics/total` operation to retrieve a count of all HTTP responses that occurred on the network in the last 30 seconds. The response body is similar to the following output:

```
{
  "stats": [
    {
      "oid": -1,
      "time": 1494541380000,
      "duration": 1800000,
      "values": [
        33357
      ]
    }
  ],
  "cycle": "30sec",
  "node_id": 0,
  "clock": 1494541440000,
  "from": 1494539640000,
  "until": 1494541440000
}
```

Operation details

POST /metrics

Specify the following parameters.

body: ***Object***

The description of the metric request.

from: ***Number***

The beginning timestamp for the request. Return only metrics collected after this time. Time is expressed in milliseconds since the epoch. 0 indicates the time of the request. A negative value is evaluated relative to the current time. The default unit for a negative value is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

until: ***Number***

The ending timestamp for the request. Return only metrics collected before this time. Follows the same time value guidelines as the from parameter.

cycle: ***String***

The aggregation period for metrics.

The following values are valid:

- `auto`
- `1sec`
- `30sec`
- `5min`
- `1hr`
- `24hr`

object_type: ***String***

Indicates the object type of unique identifiers specified in the object_ids property.

The following values are valid:

- `network`
- `device`
- `application`
- `vlan`
- `device_group`
- `system`

object_ids: ***Array of Numbers***

The list of numeric values that represent unique identifiers. Unique identifiers can be retrieved through the /networks, /devices, /applications, /vlans, /devicegroups, /activitygroups, and /appliances resources. For system health metrics, specify the ID of the sensor or console and set the object_type parameter to "system".

metric_category: ***String***

The group of metrics that are searchable in the metric catalog.

metric_specs: ***Array of Objects***

An array of metric specification objects.

name: ***String***

The field name for the metric. When filtering in the metric catalog on a metric_category, each result is a potential metric_spec name. When a result is selected from the catalog, the "Metric" field value is a valid option for this field.

key1: ***String***

(Optional) Filter detail metrics. Detail metrics break down data through keys, which are strings or IP addresses. For example, the metric "HTTP Requests by Method" accepts a key1 value of "GET." Keys can also be regular expressions that are delimited with forward slashes ("/GET/").

key2: ***String***

(Optional) Enable additional filtering on detail metrics.

calc_type: ***String***

(Optional) The type of calculation to perform.

The following values are valid:

- `mean`
- `percentiles`

percentiles: ***Array of Numbers***

(Optional) The list of percentiles, sorted in ascending order, which should be returned. This parameter is only required if the calc_type parameter is set to "percentiles". If the calc_type parameter is set to mean, the percentiles property cannot be set.

Specify the body parameter in the following JSON format.

```
{
    "cycle": "string",
    "from": 0,
    "metric_category": "string",
    "metric_specs": {
        "name": "string",
        "key1": "string",
        "key2": "string",
        "calc_type": "string",
        "percentiles": []
    },
    "object_ids": [],
    "object_type": "string",
    "until": 0
```

```
}
```

POST /metrics/total

Specify the following parameters.

body: ***Object***

The description of the metric request.

from: ***Number***

The beginning timestamp for the request. Return only metrics collected after this time. Time is expressed in milliseconds since the epoch. 0 indicates the time of the request. A negative value is evaluated relative to the current time. The default unit for a negative value is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

until: ***Number***

The ending timestamp for the request. Return only metrics collected before this time. Follows the same time value guidelines as the from parameter.

cycle: ***String***

The aggregation period for metrics.

The following values are valid:

- `auto`
- `1sec`
- `30sec`
- `5min`
- `1hr`
- `24hr`

object_type: ***String***

Indicates the object type of unique identifiers specified in the object_ids property.

The following values are valid:

- `network`
- `device`
- `application`
- `vlan`
- `device_group`
- `system`

object_ids: ***Array of Numbers***

The list of numeric values that represent unique identifiers. Unique identifiers can be retrieved through the /networks, /devices, /applications, /vlans, /devicegroups, /activitygroups, and /appliances resources. For system health metrics, specify the ID of the sensor or console and set the object_type parameter to "system".

metric_category: ***String***

The group of metrics that are searchable in the metric catalog.

metric_specs: ***Array of Objects***

An array of metric specification objects.

name: ***String***

The field name for the metric. When filtering in the metric catalog on a metric_category, each result is a potential metric_spec name. When a result is selected from the catalog, the "Metric" field value is a valid option for this field.

key1: ***String***

(Optional) Filter detail metrics. Detail metrics break down data through keys, which are strings or IP addresses. For example, the metric "HTTP Requests by Method" accepts a key1 value of "GET." Keys can also be regular expressions that are delimited with forward slashes ("/GET/").

key2: ***String***

(Optional) Enable additional filtering on detail metrics.

calc_type: ***String***

(Optional) The type of calculation to perform.

The following values are valid:

- `mean`
- `percentiles`

percentiles: ***Array of Numbers***

(Optional) The list of percentiles, sorted in ascending order, which should be returned. This parameter is only required if the calc_type parameter is set to "percentiles". If the calc_type parameter is set to mean, the percentiles property cannot be set.

Specify the body parameter in the following JSON format.

```
{
    "cycle": "string",
    "from": 0,
    "metric_category": "string",
    "metric_specs": {
        "name": "string",
        "key1": "string",
        "key2": "string",
        "calc_type": "string",
        "percentiles": []
    },
    "object_ids": [],
    "object_type": "string",
    "until": 0
}
```

POST /metrics/totalbyobject

Specify the following parameters.

body: ***Object***

The description of the metric request.

from: ***Number***

The beginning timestamp for the request. Return only metrics collected after this time. Time is expressed in milliseconds since the epoch. 0 indicates the time of the request. A negative value is evaluated relative to the current time. The default unit for a negative value is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

until: ***Number***

The ending timestamp for the request. Return only metrics collected before this time. Follows the same time value guidelines as the from parameter.

cycle: ***String***

The aggregation period for metrics.

The following values are valid:

- `auto`
- `1sec`
- `30sec`
- `5min`
- `1hr`
- `24hr`

object_type: ***String***

Indicates the object type of unique identifiers specified in the object_ids property.

The following values are valid:

- `network`
- `device`
- `application`
- `vlan`
- `device_group`
- `system`

object_ids: ***Array of Numbers***

The list of numeric values that represent unique identifiers. Unique identifiers can be retrieved through the /networks, /devices, /applications, /vlans, /devicegroups, /activitygroups, and /appliances resources. For system health metrics, specify the ID of the sensor or console and set the object_type parameter to "system".

metric_category: ***String***

The group of metrics that are searchable in the metric catalog.

metric_specs: ***Array of Objects***

An array of metric specification objects.

name: ***String***

The field name for the metric. When filtering in the metric catalog on a metric_category, each result is a potential metric_spec name. When a result is selected from the catalog, the "Metric" field value is a valid option for this field.

key1: ***String***

(Optional) Filter detail metrics. Detail metrics break down data through keys, which are strings or IP addresses. For example, the metric "HTTP Requests by Method" accepts a key1 value of "GET." Keys can also be regular expressions that are delimited with forward slashes ("/GET/").

key2: ***String***

(Optional) Enable additional filtering on detail metrics.

calc_type: ***String***

(Optional) The type of calculation to perform.

The following values are valid:

- `mean`
- `percentiles`

percentiles: ***Array of Numbers***

(Optional) The list of percentiles, sorted in ascending order, which should be returned. This parameter is only required if the calc_type parameter is set to "percentiles". If the calc_type parameter is set to mean, the percentiles property cannot be set.

Specify the body parameter in the following JSON format.

```
{
    "cycle": "string",
```

```
    "from": 0,
    "metric_category": "string",
    "metric_specs": {
        "name": "string",
        "key1": "string",
        "key2": "string",
        "calc_type": "string",
        "percentiles": []
    },
    "object_ids": [],
    "object_type": "string",
    "until": 0
}
```

GET /metrics/next/{xid}

Specify the following parameters.

xid: ***Number***

The unique identifier returned by a metric query.

Supported time units

For most parameters, the default unit for time measurement is milliseconds. However, the following parameters return or accept alternative time units such as minutes and hours:

- Device
  - active_from
  - active_until
- Device group
  - active_from
  - active_until
- Metrics
  - from
  - until
- Record Log
  - from
  - until
  - context_ttl

The following table displays supported time units:

Time unit	Unit suffix
Year	y
Month	M
Week	w
Day	d
Hour	h
Minute	m
Second	s

Time unit	Unit suffix
Millisecond	ms

To specify a time unit other than milliseconds for a parameter, append the unit suffix to the value. For example, to request devices active in the last 30 minutes, specify the following parameter value:

```
GET /api/v1/devices?active_from=-30m
```

The following example specifies a search for HTTP records created between 1 and 2 hours ago:

```
{
    "from": "-2h",
    "until": "-1h",
    "types": ["~http"]
}
```

Network

Networks are correlated to the network interface card that receives input from all of the objects identified by the ExtraHop system.

On a console, each connected sensor is identified as a network capture. For more information, see Networks.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /networks	Retrieve all networks.
GET /networks/{id}	Retreive a specific network by ID.
PATCH /networks/{id}	Update a specific network by ID.
GET /networks/{id}/alerts	Retrieve all alerts that are assigned to a specific network.
POST /networks/{id}/alerts	Assign and unassign alerts to a specific network.
DELETE /networks/{id}/alerts/{child-id}	Unassign an alert from a specific network.
POST /networks/{id}/alerts/{child-id}	Assign an alert to a specific network.
GET /networks/{id}/vlans	Retrieve all VLANS assigned to a specific network.

Operation details

GET /networks

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "appliance_uuid": "string",
    "description": "string",
    "id": 0,
    "idle": true,
    "mod_time": 0,
```

```
    "name": "string",
    "node_id": 0
}
```

PATCH /networks/{id}

Specify the following parameters.

body: ***Object***
Property value updates to apply to the network.

id: ***Number***
Unique identifier of the network.

GET /networks/{id}

Specify the following parameters.

id: ***Number***
Unique identifier of the network.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "appliance_uuid": "string",
    "description": "string",
    "id": 0,
    "idle": true,
    "mod_time": 0,
    "name": "string",
    "node_id": 0
}
```

GET /networks/{id}/alerts

Specify the following parameters.

id: ***Number***
Unique identifier of the network.

direct_assignments_only: ***Boolean***
(Optional) Restrict results to only alerts directly assigned to the network.

POST /networks/{id}/alerts

Specify the following parameters.

body: ***Object***
Lists of alert IDs to assign and/or unassign.

assign: ***Array of Numbers***
IDs of resources to assign

unassign: ***Array of Numbers***
IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
```

```
}
```

id: ***Number***
Unique identifier of the network.

POST /networks/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***
Unique identifier of the alert.

id: ***Number***
Unique identifier of the network.

DELETE /networks/{id}/alerts/{child-id}

Specify the following parameters.

child-id: ***Number***
Unique identifier of the alert.

id: ***Number***
Unique identifier of the network.

GET /networks/{id}/vlans

Specify the following parameters.

id: ***Number***
Unique identifier of the network.

Network locality entry

You can manage a list that specifies the network locality of IP addresses.

For example, you can create an entry in the network locality list that specifies that an IP address or CIDR block is internal or external.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /networklocalities	Retrieve all network locality entries.
POST /networklocalities	Create a network locality entry.
DELETE /networklocalities/{id}	Delete a network locality entry. **Note:** This operation is not available if the sensor is connected to Reveal(x) 360.
GET /networklocalities/{id}	Retrieve a specific network locality entry.
PATCH /networklocalities/{id}	Apply updates to a specific network locality entry. **Note:** This operation is not available if the sensor is connected to Reveal(x) 360.

Operation details

GET /networklocalities

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "external": true,
    "id": 0,
    "mod_time": 0,
    "network": "string"
}
```

POST /networklocalities

Specify the following parameters.

body: ***Object***

Apply the specified property values to the new network locality entry.

network: ***String***

Network CIDR block or IP address.

external: ***Boolean***

Is this network internal or external.

description: ***String***

(Optional) An optional description of the network locality entry.

Specify the body parameter in the following JSON format.

```
{
    "description": "string",
    "external": true,
    "network": "string"
}
```

GET /networklocalities/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the network locality entry.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "external": true,
    "id": 0,
    "mod_time": 0,
    "network": "string"
}
```

DELETE /networklocalities/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the network locality entry.

PATCH /networklocalities/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property value updates to the network locality entry.

id: ***Number***

The unique identifier for the network locality entry.

Node

A node is a sensor that is connected to a console.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /nodes	Retrieve all sensors connected to this console.
GET /nodes/{id}	Retrieve a specific sensor that is connected to this console.
PATCH /nodes/{id}	Update a specific sensor that is connected to this console.

Operation details

GET /nodes

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "add_time": 0,
    "display_name": "string",
    "enabled": true,
    "firmware_version": "string",
    "hostname": "string",
    "id": 0,
    "license_status": "string",
    "nickname": "string",
    "ntp_sync": true,
    "product_key": "string",
    "status_code": "string",
    "status_message": "string",
    "time_added": 0,
    "time_offset": 0,
    "uuid": "string"
}
```

GET /nodes/{id}

Specify the following parameters.

id: ***Number***

The ID of the sensor.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "add_time": 0,
    "display_name": "string",
    "enabled": true,
    "firmware_version": "string",
    "hostname": "string",
    "id": 0,
    "license_status": "string",
    "nickname": "string",
    "ntp_sync": true,
    "product_key": "string",
    "status_code": "string",
    "status_message": "string",
    "time_added": 0,
    "time_offset": 0,
    "uuid": "string"
}
```

PATCH /nodes/{id}

Specify the following parameters.

body: ***Object***

Apply the specified updates to the Discover node.

id: ***Number***

The unique identifier for the Discover node.

Observations

An observation associates the IP address of a device on the ExtraHop system with an IP address outside of your network. For example, you can track the activity of a VPN user by associating the IP address of the VPN client on your internal network with the external IP address assigned to the user on the public internet.

The following table displays all of the operations you can perform on this resource:

Operation	Description
POST /observations/associatedipaddrs	Add an observation to create an association between device IP addresses.

Operation details

POST /observations/associatedipaddrs

Specify the following parameters.

body: ***Object***

The observation parameters.

observations: ***Array of Objects***

An array of observations.

ipaddr: ***String***

The device IP address observed by the sensor or console.

associated_ipaddr: ***String***

The associated IP address.

timestamp: ***Number***

The time that the observation was created by the source, expressed in milliseconds since the epoch.

source: ***String***

The source of the observations.

Specify the body parameter in the following JSON format.

```
{
    "observations": {
        "ipaddr": "string",
        "associated_ipaddr": "string",
        "timestamp": 0
    },
    "source": "string"
}
```

Open Data Stream

An open data stream (ODS) is a channel through which you can send specified metric data from a sensor to an external, third-party system. For example, you might want to store or analyze metric data with a remote tool, such as Splunk, MongoDB, or Amazon Web Services (AWS).

Sending data through an open data stream is a two-step procedure. First, you configure a connection to the target system that will receive the data. Second, you write a trigger that specifies what data to send to the target system and when to send it. For more information, see Open Data Streams.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /odstargets	Retrieve all Open Data Stream targets.
GET /odstargets/http	Retrieve all HTTP Open Data Stream targets.
POST /odstargets/http	Create a new HTTP Open Data Stream target.
DELETE /odstargets/http/{name}	Delete an HTTP Open Data Stream target.
GET /odstargets/http/{name}	Retrieve a specific HTTP Open Data Stream target.
GET /odstargets/kafka	Retrieve all Kafka Open Data Stream targets.
POST /odstargets/kafka	Create a new Kafka Open Data Stream target.
DELETE /odstargets/kafka/{name}	Delete a Kafka Open Data Stream target.
GET /odstargets/kafka/{name}	Retrieve a specific Kafka Open Data Stream target.
GET /odstargets/mongodb	Retrieve all MongoDB Open Data Stream targets.
POST /odstargets/mongodb	Create a new MongoDB Open Data Stream target.
DELETE /odstargets/mongodb/{name}	Delete a MongoDB Open Data Stream target.
GET /odstargets/mongodb/{name}	Retrieve a specific MongoDB Open Data Stream target.

Operation	Description
GET /odstargets/raw	Retrieve all Raw Open Data Stream targets.
POST /odstargets/raw	Create a new Raw Open Data Stream target.
DELETE /odstargets/raw/{name}	Delete a Raw Open Data Stream target.
GET /odstargets/raw/{name}	Retrieve a specific Raw Open Data Stream target.
GET /odstargets/syslog	Retrieve all Syslog Open Data Stream targets.
POST /odstargets/syslog	Create a new Syslog Open Data Stream target.
DELETE /odstargets/syslog/{name}	Delete a Syslog Open Data Stream target.
GET /odstargets/syslog/{name}	Retrieve a specific Syslog Open Data Stream target.

Operation details

GET /odstargets

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/http

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/http/{name}

Specify the following parameters.

name: ***String***

The name of the target.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/kafka

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/kafka/{name}

Specify the following parameters.

name: ***String***

The name of the target.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/mongodb

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/mongodb/{name}

Specify the following parameters.

name: ***String***
The name of the target.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/raw

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/raw/{name}

Specify the following parameters.

name: ***String***
The name of the target.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/syslog

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

GET /odstargets/syslog/{name}

Specify the following parameters.

name: ***String***
The name of the target.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{}
```

POST /odstargets/http

Specify the following parameters.

body: ***Object***

name: ***String***

The name for the target.

host: ***String***

The hostname or IP address of the remote HTTP server.

port: ***Number***

The TCP port number of the HTTP server.

protocol: ***String***

The protocol to transmit data over.

The following values are valid:

- `http`
- `https`

skip_cert_verification: ***Boolean***

(Optional) Indicates whether to bypass TLS certificate verification for encrypted data. This parameter is valid only if `protocol` is set to `https`.

pipeline: ***Boolean***

Indicates whether multiple concurrent HTTP connections are enabled, which can improve throughput speed.

additional_header: ***String***

(Optional) Specifies an additional HTTP header to include in each request. Headers must be specified in the following format: "<key>:<value>". For example: "additional_header": "Accept: text/html".

authentication: ***Object***

An object that contains HTTP authentication credentials.

auth_type: ***String***

The type of HTTP authentication.

The following values are valid:

- `none`
- `basic`
- `aws`
- `azure_storage`
- `azure_ad`
- `crowdstrike`

username: ***String***

(Optional) The name of the user. This option is required if `auth_type` is set to `basic` or if `auth_type` is set to `azure_ad` and `grant_type` is set to `resource_owner`.

password: ***String***

(Optional) The password of the user. This option is required if `auth_type` is set to `basic` or if `auth_type` is set to `azure_ad` and `grant_type` is set to `resource_owner`.

access_key: ***String***

(Optional) The access key ID. This option is required for AWS and Azure Storage authentication.

secret_key: ***String***

(Optional) The secret access key. This option is required for AWS authentication.

service: ***String***

(Optional) The service code of the AWS service, such as "AmazonEC2". This option is required for AWS authentication.

region: ***String***

(Optional) The name of the AWS region, such as "us-west-1". This option is required for AWS authentication.

grant_type: ***String***

(Optional) The OAuth 2.0 grant type. This option is required for Azure AD authentication.

The following values are valid:

- `client`
- `resource_owner`

client_id: ***String***

(Optional) The client ID. This option is required for Azure AD and Crowdstrike authentication.

client_secret: ***String***

(Optional) The client Secret Key. This option is required for Azure AD and Crowdstrike authentication.

resource: ***String***

(Optional) The Azure AD resource URI. This option is required for Azure AD authentication.

token_endpoint: ***String***

(Optional) The Azure AD /token endpoint. For example: "https://login.microsoftonline.com/<tenant_id>/oauth2/token". This option is required for Azure AD authentication.

Specify the body parameter in the following JSON format.

```
{
    "additional_header": "string",
    "authentication": {
        "auth_type": "string",
        "username": "string",
        "password": "string",
        "access_key": "string",
        "secret_key": "string",
        "service": "string",
        "region": "string",
        "grant_type": "string",
        "client_id": "string",
        "client_secret": "string",
        "resource": "string",
        "token_endpoint": "string"
    },
    "host": "string",
    "name": "string",
    "pipeline": true,
    "port": 0,
    "protocol": "string",
    "skip_cert_verification": true
}
```

POST /odstargets/kafka

Specify the following parameters.

body: ***Object***

name: ***String***

The name for the target.

brokers: ***Array of Objects***

An array of one or more objects that contain information about Kafka Brokers.

host: ***String***

The hostname or IP address of the remote Kafka broker.

port: ***Number***

The TCP port number of the Kafka broker.

compression: ***String***

(Optional) The compression method to apply to transmitted data.

The following values are valid:

- `none`
- `gzip`
- `snappy`

partition_strategy: ***String***

(Optional) The partitioning method to apply to transmitted data.

The following values are valid:

- `hash_key`
- `manual`
- `random`
- `round_robin`

Specify the body parameter in the following JSON format.

```
{
    "brokers": {
        "host": "string",
        "port": 0
    },
    "compression": "string",
    "name": "string",
    "partition_strategy": "string"
}
```

POST /odstargets/mongodb

Specify the following parameters.

body: ***Object***

name: ***String***

The name for the target.

host: ***String***

The hostname or IP address of the remote MongoDB server.

port: ***Number***

The TCP port number of the MongoDB server.

encrypt: ***Boolean***

(Optional) Indicates whether data is encrypted with TLS.

skip_cert_verification: ***Boolean***

(Optional) Indicates whether to bypass TLS certificate verification for encrypted data. This parameter is valid only if `encrypt` is set to `true`.

authentication: ***Array of Objects***

(Optional) An array of objects that contain MongoDB authentication credentials.

database: ***String***

The name of the MongoDB database.

user: ***String***

The name of the user that has permission to modify the specified database.

password: ***String***

The password of the user.

Specify the body parameter in the following JSON format.

```
{
    "authentication": {
        "database": "string",
        "user": "string",
        "password": "string"
    },
    "encrypt": true,
    "host": "string",
    "name": "string",
    "port": 0,
    "skip_cert_verification": true
}
```

POST /odstargets/raw

Specify the following parameters.

body: ***Object***

name: ***String***

The name for the target.

host: ***String***

The hostname or IP address of the remote server.

port: ***Number***

The TCP or UDP port number of the remote server.

protocol: ***String***

The protocol to transmit data over.

The following values are valid:

- `tcp`
- `udp`

compression: ***Boolean***

(Optional) Indicates whether gzip compression is applied to transmitted data.

gzip_threshold_bytes: ***Number***

(Optional) The number of bytes that specifies the threshold for creating a new message. Every 30 seconds, the sensor or console sends messages that exceed the specified size to prevent messages from growing too large. This option is valid only if `compression` is set to `true`.

gzip_threshold_seconds: ***Number***

(Optional) The number of seconds that specifies the threshold for creating a new message. Every 30 seconds, the sensor or console sends messages that have been written for more than the specified time period to prevent messages from growing too large. This option is valid only if `compression` is set to `true`.

Specify the body parameter in the following JSON format.

```
{
    "compression": true,
    "gzip_threshold_bytes": 0,
    "gzip_threshold_seconds": 0,
    "host": "string",
    "name": "string",
    "port": 0,
    "protocol": "string"
}
```

POST /odstargets/syslog

Specify the following parameters.

body: ***Object***

name: ***String***

The name for the target.

host: ***String***

The hostname or IP address of the remote Syslog server.

port: ***Number***

The TCP or UDP port number of the remote Syslog server.

protocol: ***String***

The protocol to transmit data over.

The following values are valid:

- `tcp`
- `udp`
- `tls`

localtime: ***Boolean***

(Optional) Indicates whether remote syslog information is sent with timestamps in the local time zone of the sensor or console. If this parameter is set to `false`, timestamps are sent in GMT.

tls_client_cert: ***String***

(Optional) The TLS client certificate that is sent to the Syslog server during the TLS handshake. Specify this option if the Syslog server uses client authentication.

tls_client_key: ***String***

(Optional) The private key of the TLS client certificate specified by the tls_client_cert parameter. Specify this option if the Syslog server uses client authentication.

tls_ca_certs: ***String***

(Optional) The trusted certificates to validate the Syslog server certificate with, in PEM format. Specify this option if your Syslog server certificate has not been signed by a valid Certificate Authority (CA). If this option is not specified, the server certificate is validated with the built-in list of valid CA certificates. This option is valid only if the protocol is TLS.

Specify the body parameter in the following JSON format.

```
{
    "host": "string",
    "localtime": true,
    "name": "string",
    "port": 0,
    "protocol": "string",
    "tls_ca_certs": "string",
    "tls_client_cert": "string",
    "tls_client_key": "string"
}
```

DELETE /odstargets/http/{name}

Specify the following parameters.

name: ***String***

The name of the target.

DELETE /odstargets/kafka/{name}

Specify the following parameters.

name: ***String***

The name of the target.

DELETE /odstargets/mongodb/{name}

Specify the following parameters.

name: ***String***

The name of the target.

DELETE /odstargets/raw/{name}

Specify the following parameters.

name: ***String***

The name of the target.

DELETE /odstargets/syslog/{name}

Specify the following parameters.

name: ***String***

The name of the target.

Packet capture

You can retrieve and delete packets stored on the ExtraHop system.

Note: For ExtraHop Reveal(x), this resource is not supported and has been replaced by the Packet Search resource.

You must write a trigger to identify the information you want to generate. For example, you can write a trigger to collect all of the packets going to a particular device that is generating a high volume of errors. Then, you can download or delete that information. For more information, see Packets.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /packetcaptures	Retrieve metadata about all packet captures stored on this ExtraHop system.
DELETE /packetcaptures/{id}	Permanently remove a specific packet capture from the ExtraHop system.
GET /packetcaptures/{id}	Download a specific packet capture in PCAP format.

Operation details

GET /packetcaptures

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "bytes": 0,
    "device_id1": "string",
    "device_id2": "string",
    "id": "string",
    "ipaddr1": "string",
    "ipaddr2": "string",
    "ipproto": 0,
    "l7proto": "string",
    "name": "string",
    "pkts": 0,
    "port1": 0,
    "port2": 0,
    "snaplen": 0,
    "tend_usec": 0,
    "tstart_usec": 0,
    "vlanid": 0
}
```

GET /packetcaptures/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the packet capture.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "bytes": 0,
    "device_id1": "string",
    "device_id2": "string",
    "id": "string",
    "ipaddr1": "string",
    "ipaddr2": "string",
    "ipproto": 0,
    "l7proto": "string",
    "name": "string",
    "pkts": 0,
    "port1": 0,
    "port2": 0,
    "snaplen": 0,
```

```
    "tend_usec": 0,
    "tstart_usec": 0,
    "vlanid": 0
}
```

DELETE /packetcaptures/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the packet capture.

Packet Search

You can search for and download packets stored on the ExtraHop system. The downloaded packets can then be analyzed through a third-party tool, such as Wireshark.

Note: This resource can only retrieve packets from packetstores. To retrieve packets from a packet sensor, see the Packet Capture resource.

For more information about Packets, see Packets.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /packets/search	Search for packets by specifying parameters in a URL.
POST /packets/search	Search for packets by specifying parameters in a JSON string.

Operation details

GET /packets/search

Specify the following parameters.

output: ***String***

(Optional) The output format. * `pcap` - A PCAP file that contains packets. * `keylog_txt` - A keylog text file that contains secrets for decryption. * `pcapng` - A PCAPNG file that can contain both packets and secrets for decryption. * `zip` - A ZIP file that contains both a PCAP and keylog text file.

The following values are valid:

- `pcap`
- `keylog_txt`
- `pcapng`
- `zip`

include_secrets: ***Boolean***

(Optional) Specifies whether to include secrets in the PCAPNG file. This option is valid only if `output` is set to `pcapng`.

limit_bytes: ***String***

(Optional) The maximum number of bytes to return.

limit_search_duration: ***String***

(Optional) The maximum amount of time to run the packet search. The default unit is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

always_return_body: ***Boolean***

(Optional) If you set this parameter to true, and the search does not match any packets, the system returns an empty packet capture file and an HTTP status of 200. If you set this parameter to false, and the search does not match any packets, the system returns no packet capture file and an HTTP status of 204.

from: ***String***

The beginning timestamp of the time range the search will include, expressed in milliseconds since the epoch. A negative value specifies that the search will begin with packets captured at a time in the past. For example, specify -10m to begin the search with packets captured 10 minutes before the time of the request. Negative values can be specified with a time unit other than milliseconds, such as seconds or hours. See the REST API Guide for supported time units and suffixes.

until: ***String***

(Optional) The ending timestamp of the time range the search will include, expressed in milliseconds since the epoch. A 0 value specifies that the search will end with packets captured at the time of the search. A negative value specifies that the search will end with packets captured at a time in the past. For example, specify -5m to end the search with packets captured 5 minutes before the time of the request. Negative values can be specified with a time unit other than milliseconds, such as seconds or hours. See the REST API Guide for supported time units and suffixes.

bpf: ***String***

(Optional) The Berkeley Packet Filter (BPF) syntax for the packet search. For more information about BPF syntax, see the REST API Guide.

ip1: ***String***

(Optional) Returns packets sent to or received by the specified IP address.

port1: ***String***

(Optional) Returns packets sent from or received on the specified port.

ip2: ***String***

(Optional) Returns packets sent to or received by the specified IP address.

port2: ***String***

(Optional) Returns packets sent from or received on the specified port.

POST /packets/search

Specify the following parameters.

body: ***Object***

The parameters of the packet search.

output: ***String***

(Optional) The output format.

The following values are valid:

- `pcap`
- `keylog_txt`
- `pcapng`
- `zip`

include_secrets: ***Boolean***

(Optional) Whether or not to include TLS secrets together with packet data in .pcapng files. Only valid if "output" is "pcapng".

limit_bytes: ***String***

(Optional) The maximum number of bytes to return.

limit_search_duration: ***String***

(Optional) The maximum amount of time to run the packet search. The default unit is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes.

always_return_body: ***Boolean***

(Optional) If you set this parameter to true, and the search does not match any packets, the system returns an empty packet capture file and an HTTP status of 200. If you set this parameter to false, and the search does not match any packets, the system returns no packet capture file and an HTTP status of 204.

from: ***String***

The beginning timestamp of the time range the search will include, expressed in milliseconds since the epoch. A negative value specifies that the search will begin with packets captured at a time in the past. For example, specify -10m to begin the search with packets captured 10 minutes before the time of the request. Negative values can be specified with a time unit other than milliseconds, such as seconds or hours. See the REST API Guide for supported time units and suffixes.

until: ***String***

(Optional) The ending timestamp of the time range the search will include, expressed in milliseconds since the epoch. A 0 value specifies that the search will end with packets captured at the time of the search. A negative value specifies that the search will end with packets captured at a time in the past. For example, specify -5m to end the search with packets captured 5 minutes before the time of the request. Negative values can be specified with a time unit other than milliseconds, such as seconds or hours. See the REST API Guide for supported time units and suffixes.

bpf: ***String***

(Optional) The Berkeley Packet Filter (BPF) syntax for the packet search. For more information about BPF syntax, see Filter packets with Berkeley Packet Filter syntax.

ip1: ***String***

(Optional) Returns packets sent to or received by the specified IP address.

port1: ***String***

(Optional) Returns packets sent from or received on the specified port.

ip2: ***String***

(Optional) Returns packets sent to or received by the specified IP address.

port2: ***String***

(Optional) Returns packets sent from or received on the specified port.

Specify the body parameter in the following JSON format.

```
{
    "always_return_body": true,
    "bpf": "string",
    "from": "string",
    "include_secrets": true,
    "ip1": "string",
    "ip2": "string",
    "limit_bytes": "string",
    "limit_search_duration": "string",
```

```
    "output": "string",
    "port1": "string",
    "port2": "string",
    "until": "string"
}
```

Filter packets with Berkeley Packet Filter syntax

Search for packets with the Berkeley Packet Filter (BPF) syntax alone, or in combination with the built-in filters.

Berkeley Packet Filters are a raw interface to data link layers and are a powerful tool for intrusion detection analysis. The BPF syntax enables users to write filters that quickly drill down on specific packets to see the essential information.

The ExtraHop system constructs a synthetic packet header from the packet index data and then runs the BPF syntax queries against the packet header to ensure that queries are much faster than scanning the full packet payload. Note that ExtraHop supports only a subset of the BPF syntax. See Supported BPF syntax.

The BPF syntax consists of one or more primitives preceded by one or more qualifiers. Primitives usually consist of an ID (name or number) preceded by one or more qualifiers. There are three different kinds of qualifiers:

type

Qualifiers that indicate what type the ID name or number refers to. For example, `host`, `net`, `port`, and `portrange`. If there is no qualifier, `host` is assumed.

dir

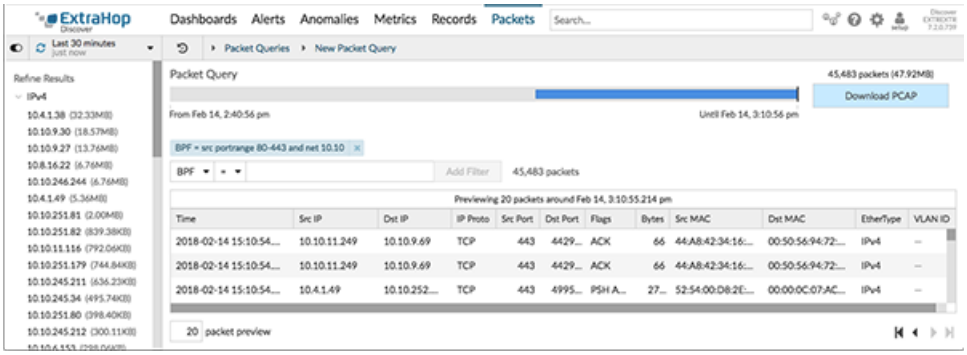
Qualifiers that specify a particular transfer direction to and or from an ID. Possible directions are `src`, `dst`, `src and dst`, and `src or dst`. For example, `dst net 128.3`.

proto

Qualifiers that restrict the match to the particular protocol. Possible protocols are `ether`, `ip`, `ip6`, `tcp`, and `udp`.

Add a filter with BPF syntax

1. Log in to the ExtraHop system through `https://<extrahop-hostname-or-IP-address>`.
2. From the top menu, click **Packets**.
3. In the trifield filter section, select **BPF**, and then type your filter syntax. For example, type `src portrange 80-443 and net 10.10`.
4. Click **Download PCAP** to save the packet capture with your filtered results.



Supported BPF syntax

The ExtraHop system supports the following subset of the BPF syntax for filtering packets.

Note:
- ExtraHop only supports numeric IP address searches. Hostnames are not allowed.
- Indexing into headers, `[...]`, is only supported for `tcpflags` and `ip_offset`. For example, `tcp[tcpflags] & (tcp-syn|tcp-fin) != 0`

- ExtraHop supports both numeric and hexadecimal values for VLAN ID, EtherType, and IP Protocol fields. Prefix hexadecimal values with 0x, such as 0x11.

Primitive	Examples	Description
`[src|dst] host <host ip>`	`host 203.0.113.50` `dst host 198.51.100.200`	Matches a host as the IP source, destination, or either. These host expressions can be specified in conjunction with other protocols like ip, arp, rarp or ip6.
`ether [src|dst] host <MAC>`	`ether host 00:00:5E:00:53:00` `ether dst host 00:00:5E:00:53:00`	Matches a host as the Ethernet source, destination, or either.
`vlan <ID>`	`vlan 100`	Matches a VLAN. Valid ID numbers are `0-4095`. VLAN priority bits are zero. If the original packet had more than one VLAN tag, the synthetic packet the BPF matches against will only have the innermost VLAN tag.

`[src|dst] portrange <p1>-<p2>`
or
`[tcp|udp] [src|dst] portrange <p1>-<p2>`
	`src portrange 80-88`
	`tcp dst portrange 1501-1549`
		Matches packets to or from a port in the given range. Protocols can be applied to a port range to filter specific packets within the range.

`[ip|ip6][src|dst] proto <protocol>`	`proto 1`
	`src 10.4.9.40 and proto ICMP`
	`ip6 and src fe80::aebc:32ff:fe84:70b7 and proto 47`
	`ip and src 10.4.9.40 and proto 0x0006`
		Matches IPv4 or IPv6 protocols other than TCP and UDP. The protocol can be a number or name.

`[ip|ip6][tcp|udp] [src|dst] port <port>`	`udp and src port 2005` `ip6 and tcp and src port 80`	Matches IPv4 or IPv6 packets on a specific port.

`[src|dst] net <network>`	`dst net 192.168.1.0`
	`src net 10`
	`net 192.168.1.0/24`
		Matches packets to or from a source or destination or either, that reside in a network. An IPv4 network number can be specified as one of the following values:
		• Dotted quad (x.x.x.x)
		• Dotted triple (x.x.x)
		• Dotted pair (x.x)
		• Single number (x)

Primitive	Examples	Description

`[ip|ip6] tcp tcpflags & (tcp-[ack|fin|syn|rst|push|urg|)`	`tcp[tcpflags] & (tcp-ack) !=0`
	`tcp[13] & 16 !=0`
	`ip6 and (ip6[40+13] & (tcp-syn) != 0)`
		Matches all packets with the specified TCP flag

Fragmented IPv4 packets (ip_offset != 0)	`ip[6:2] & 0x3fff != 0x0000`	Matches all packets with fragments.

Pairing

This resource enables you to generate a token required to connect a sensor to a console.

The following table displays all of the operations you can perform on this resource:

Operation	Description
POST /pairing/token	Generate a token required to connect the sensor to a console.

Operation details

POST /pairing/token

There are no parameters for this operation.

Record Log

Records are structured flow and transaction information about events on your network.

After you connect the ExtraHop system to a record store, you can generate and send record information to the recordstore, and you can query records to retrieve information about any object on your network. For more information, see Query for records through the REST API.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /records/cursor/{cursor}	Deprecated. Replaced by `POST /records/cursor`.
POST /records/cursor	Retrieve records starting at a specified cursor.
POST /records/search	Perform a record log query.

Operation details

POST /records/search

Specify the following parameters.

body: ***Object***

The record log query.

from: ***Number***

The beginning timestamp of the time range the query will search, expressed in milliseconds since the epoch. A negative value specifies that the search will begin with records created at a time in the past. For example, specify -600000ms to begin the search with records created 10 minutes before the time of the request. Negative values can be specified with a time unit other than milliseconds, such as seconds or hours. See the REST API Guide for supported time units and suffixes.

until: ***Number***

The ending timestamp of the time range the query will search, expressed in milliseconds since the epoch. A 0 value specifies that the search will end with records created at the time of the request. A negative value specifies that the search will end with records created at a time in the past. For example, specify -300000ms to end the search with records created 5 minutes before the time of the request. Negative values can be specified with a time unit other than milliseconds, such as seconds or hours. See the REST API Guide for supported time units and suffixes.

types: ***Array of Strings***

(Optional) An array of one or more record formats. The query returns only records that match the specified formats. If no value is specified, the query returns records of any type. Valid values for this field are displayed in the Record Type field on the Record Formats page. For example: "~cifs".

limit: ***Number***

The maximum number of records returned by the query. The maximum value cannot exceed 10000. The default value is 100.

offset: ***Number***

The number of records to skip in the query results. The query will return records starting from the offset value. This parameter is often combined with the limit and sort parameters. The default value is 0. For ExtraHop recordstores, the maximum value is 10,000; to retrieve records returned after the first 10,000, see POST /records/cursor/. For third-party recordstores, there is no maximum value.

sort: ***Array of Objects***

The list of one or more sort objects that specify sort priorities. The returned records are sorted in the order the objects are listed. The parameters are defined under the sort_item section below. If no sort_item values are provided, records are sorted by timestamp in descending order.

field: ***String***

The field name that returned records are sorted by.

direction: ***String***

The order in which returned records are sorted. The default order is descending. After all other sorting criteria are applied, or if no sorting criteria was specified, the default order is descending by timestamp.

The following values are valid:

- `asc`
- `desc`

filter: ***Object***

The object containing the parameters that specify the filter criteria. The parameters are defined under the filter section below. If no filter values are provided, the query returns all records that match the time range and any specified record formats.

field: ***String***

The name of the field in the record to be filtered. The query compares the contents of the field parameter to the value of the operand parameter. If the specified field

name is ".any", the union of all field values will be searched. If the specified field name is ".ipaddr" or ".port", the client, server, sender, and receiver roles are included in the search. Field names are located in record formats that can be viewed in the ExtraHop system.

operator: ***String***

The compare method applied when matching the operand value against the field contents. All filter objects require an operator.

The following values are valid:

- `>`
- `<`
- `<=`
- `>=`
- `=`
- `!=`
- `startswith`
- `~`
- `!~`
- `and`
- `or`
- `not`
- `exists`
- `not_exists`
- `in`
- `not_in`

operand: ***String or Number or Object***

The value that the query attempts to match. The query compares the value of the operand to the contents of the field parameter and applies the compare method specified by the operator parameter. You can explicitly specify the operand data type as described in the REST API Guide.

rules: ***Array of Objects***

The list of one or more filter objects within a single filter object. Filter objects can be embedded recursively. Only "and", "or", and "not" operators are allowed for this parameter.

context_ttl: ***Number***

The amount of time to keep the search context active. The specified value is interpreted as a duration into the future. The default unit is milliseconds, but other units can be specified with a unit suffix. See the REST API Guide for supported time units and suffixes. If a non-null value is specified, the response includes a cursor ID that is accepted by POST /records/cursor/. This parameter is not supported for third-party recordstores.

Specify the body parameter in the following JSON format.

```
{
    "context_ttl": 0,
    "filter": {
        "field": "string",
        "operator": "string",
        "operand": "string",
        "rules": []
    },
    "from": 0,
    "limit": 0,
    "offset": 0,
```

```
    "sort": {
        "field": "string",
        "direction": "string"
    },
    "types": [],
    "until": 0
}
```

POST /records/cursor

Specify the following parameters.

body: ***Object***

The cursor ID that specifies the next page of results in the query.

cursor: ***String***

The unique identifier of the cursor that specifies the next page of results in the query.

Specify the body parameter in the following JSON format.

```
{
    "cursor": "string"
}
```

context_ttl: ***Number***

(Optional) The amount of time to keep the search context active, expressed in milliseconds.

GET /records/cursor/{cursor}

Specify the following parameters.

cursor: ***String***

The cursor ID.

context_ttl: ***Number***

(Optional) The amount of time to keep the search context active, expressed in milliseconds.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "cursor": "string",
    "from": 0,
    "records": {},
    "total": 0,
    "until": 0,
    "warnings": {}
}
```

Operand values in record queries

The `operand` field in the `POST /records/search` method specifies the value that a record query attempts to match. You can specify either the value only or both the data type and the value. If you specify only the value, the query will refer to the record format associated with the `field` parameter to determine the data type of the value.

For example, if you want to search for an IP address, you can specify an IP address data type, and then provide the actual address as the value.

The following example explicitly specifies the data type and value of the operand:

```
{
    "from": -1000,
```

```
    "filter": {
        "field" : "senderAddr",
        "operator": "=",
        "operand" : { "type" : "ipaddr4", "value": "1.2.3.4" }
    }
}
```

The following example specifies only the value of the operand:

```
{
    "from": -1000,
    "filter": {
        "field" : "senderAddr",
        "operator": "=",
        "operand" : "1.2.3.4"
    }
}
```

You can explicitly specify the following data types in the `operand` field:

- application
- boolean
- device

  Note: You must specify the discovery ID of the device in the value field. You can find the discovery ID of a device through the `GET /devices` method.
- device_filter
- device_group
- flowinterface
- flownetwork
- ipaddr4
- ipaddr6
- number
- object
- string

The `operand` field supports CIDR notation when filtering by IP addresses; the `operator` field must be set to "=" or "!=".

You can specify multiple filters by including the `rules` option, as shown in the following example:

```
{
  "filter": {
    "operator": "and",
    "rules": [
      {
        "field": "method",
        "operand": "SMB2_READ",
        "operator": "="
      },
      {
        "field": "reqL2Bytes",
        "operand": "100",
        "operator": ">"
      }
    ]
  },
  "types": [
    "~cifs"
  ],
    "from": "-30m"
```

```
}
```

Query records with a device group filter

To filter records by device group in the REST API, you must send a POST request to the `/records/search` endpoint with a record query filter that meets the following criteria:

- The `field` must specify devices, such as `client`, `server`, `sender`, or `receiver`.
- The `operator` must be either `in` or `not_in`.
- The `operandtype` must be `device_group`.
- The `operandvalue` must be a string representation of the numerical device group ID. You can retrieve device group IDs by running the GET /devicegroup operation and viewing the contents of the `id` field in the response.

For example, the following query searches for records in which the client device was a member of a device group with an ID of 200:

```
{
    "from": "-30m",
    "filter": {
        "field": "client",
        "operator": "in",
        "operand": {
            "type": "device_group",
            "value": "200"
        }
    }
}
```

You can also filter records by device group criteria without creating a device group by specifying the operand type as `device_filter`. For example, the following query searches for records in which the client device is running Windows 10:

```
{
    "from": "-30m",
    "filter": {
        "field": "client",
        "operator": "in",
        "operand": {
            "type": "device_filter",
            "value": {
                "field": "software",
                "operand": "windows_10",
                "operator": "="
            }
        }
    }
}
```

Note: Operand values with type `device_filter` for record search are formatted the same as device search filters. For more information, see Operand values for device groups.

Supported time units

For most parameters, the default unit for time measurement is milliseconds. However, the following parameters return or accept alternative time units such as minutes and hours:

- Device
  - active_from
  - active_until

- Device group
  - active_from
  - active_until
- Metrics
  - from
  - until
- Record Log
  - from
  - until
  - context_ttl

The following table displays supported time units:

Time unit	Unit suffix
Year	`y`
Month	`M`
Week	`w`
Day	`d`
Hour	`h`
Minute	`m`
Second	`s`
Millisecond	`ms`

To specify a time unit other than milliseconds for a parameter, append the unit suffix to the value. For example, to request devices active in the last 30 minutes, specify the following parameter value:

```
GET /api/v1/devices?active_from=-30m
```

The following example specifies a search for HTTP records created between 1 and 2 hours ago:

```
{
    "from": "-2h",
    "until": "-1h",
    "types": ["~http"]
}
```

Report

A report is a PDF file of a dashboard that you can schedule for email delivery to one or more recipients. You can specify how often the report email is delivered and the time interval for dashboard data included in the PDF file.

Important: You can only schedule reports from an ECA VM.

Here are some important considerations about scheduled reports:

- You can only create a report for dashboards that you own or have been shared with you. Your ability to create a report is determined by your user privileges. Contact your ExtraHop administrator for help.
- Each report can only link to one dashboard.

- If you created a report for a dashboard that was later deleted or became inaccessible to you, the scheduled email will continue to be sent to recipients. However, the email will not include the PDF file and will instead notify recipients that the dashboard is unavailable to the report owner.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /reports	Retrieve all reports.
POST /reports	Create a report.
DELETE /reports/{id}	Delete a specific report.
GET /reports/{id}	Retrieve a specific report.
PATCH /reports/{id}	Update a specific report.
GET /reports/{id}/contents	Retrieve the contents of a specific report.
PUT /reports/{id}/contents	Replace the contents of a specific report.
GET /reports/{id}/download	Retrieve the PDF of a report.
POST /reports/{id}/emailgroups	Change the email group assigned to a specific scheduled report.
GET /reports/{id}/emailgroups	Retrieve a list of email groups assigned to a specific scheduled report.
DELETE /reports/{id}/emailgroups/{group-id}	Remove an email group from a specific scheduled report.
POST /reports/{id}/emailgroups/{group-id}	Add an email group to a specific scheduled report.
POST /reports/{id}/queue	Immediately generate and send a specific report.

Operation details

GET /reports

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "cc": [],
    "description": "string",
    "email_message": "string",
    "email_subject": "string",
    "enabled": true,
    "from": "string",
    "id": 0,
    "include_links": "string",
    "name": "string",
    "output": {},
    "owner": "string",
    "schedule": {},
    "until": "string"
}
```

POST /reports

Specify the following parameters.

body: ***Object***

The contents of the report.

name: ***String***

The name of the report.

description: ***String***

(Optional) The description of the report.

owner: ***String***

The username of the report owner.

cc: ***Array of Strings***

The list of email addresses, not included in an email group, to receive reports.

enabled: ***Boolean***

(Optional) Indicates whether the report is enabled.

from: ***String***

The beginning timestamp of the time interval for the report contents, relative to the current time and expressed in milliseconds.

until: ***String***

(Optional) The ending timestamp of the time interval for the report contents, relative to the current time and expressed in milliseconds.

email_subject: ***String***

(Optional) The content of the subject line for the report email.

schedule: ***Object***

(Optional) The object containing the parameters that specify the scheduled time range to generate and send the report. The parameters are defined in the schedule_type section below.

type: ***String***

The type of delivery schedule for the report.

The following values are valid:

- `hourly`
- `daily`
- `weekly`

at: ***Array of Objects***

(Optional) The list of objects that specify the delivery parameters for the report. The parameters are defined in the at_type section below.

by_day: ***Array of Strings***

(Optional) The days of the week to send the report.

The following values are valid:

- `mo`
- `tu`
- `we`
- `th`
- `fr`
- `sa`
- `su`

tz: ***String***

(Optional) The timezone in which to send the report.

hour: ***Number***

(Optional) The hour at which to send the report.

minute: ***Number***

(Optional) The minute at which to send the report.

output: ***Object***

The object containing the parameters that specify the output format for the report. The parameters are defined in the format_type section below.

type: ***String***

The output format for the report.

The following values are valid:

- `pdf`

width: ***String***

(Optional) The width option for the report output.

The following values are valid:

- `narrow`
- `medium`
- `wide`

pagination: ***String***

(Optional) The pagination scheme for the report output.

The following values are valid:

- `per_region`

theme: ***String***

(Optional) The display theme for the report output.

The following values are valid:

- `light`
- `dark`
- `space`
- `contrast`

Specify the body parameter in the following JSON format.

```
{
    "cc": [],
    "description": "string",
    "email_subject": "string",
    "enabled": true,
    "from": "string",
    "name": "string",
    "output": {
        "type": "string",
        "width": "string",
        "pagination": "string",
        "theme": "string"
    },
    "owner": "string",
    "schedule": {
        "type": "string",
        "at": {
            "by_day": [],
            "tz": "string",
            "hour": 0,
            "minute": 0
        }
    },
```

```
    "until": "string"
}
```

POST /reports/{id}/queue

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

PATCH /reports/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

body: ***Object***
The contents of the report.

name: ***String***
The name of the report.

description: ***String***
(Optional) The description of the report.

owner: ***String***
The username of the report owner.

cc: ***Array of Strings***
The list of email addresses, not included in an email group, to receive reports.

enabled: ***Boolean***
(Optional) Indicates whether the report is enabled.

from: ***String***
The beginning timestamp of the time interval for the report contents, relative to the current time and expressed in milliseconds.

until: ***String***
(Optional) The ending timestamp of the time interval for the report contents, relative to the current time and expressed in milliseconds.

email_subject: ***String***
(Optional) The content of the subject line for the report email.

schedule: ***Object***
(Optional) The object containing the parameters that specify the scheduled time range to generate and send the report. The parameters are defined in the schedule_type section below.

type: ***String***
The type of delivery schedule for the report.

The following values are valid:

- `hourly`
- `daily`
- `weekly`

at: ***Array of Objects***
(Optional) The list of objects that specify the delivery parameters for the report. The parameters are defined in the at_type section below.

by_day: ***Array of Strings***

(Optional) The days of the week to send the report.

The following values are valid:

- `mo`
- `tu`
- `we`
- `th`
- `fr`
- `sa`
- `su`

tz: ***String***

(Optional) The timezone in which to send the report.

hour: ***Number***

(Optional) The hour at which to send the report.

minute: ***Number***

(Optional) The minute at which to send the report.

output: ***Object***

The object containing the parameters that specify the output format for the report. The parameters are defined in the format_type section below.

type: ***String***

The output format for the report.

The following values are valid:

- `pdf`

width: ***String***

(Optional) The width option for the report output.

The following values are valid:

- `narrow`
- `medium`
- `wide`

pagination: ***String***

(Optional) The pagination scheme for the report output.

The following values are valid:

- `per_region`

theme: ***String***

(Optional) The display theme for the report output.

The following values are valid:

- `light`
- `dark`
- `space`
- `contrast`

Specify the body parameter in the following JSON format.

```
{
    "cc": [],
    "description": "string",
```

```
    "email_subject": "string",
    "enabled": true,
    "from": "string",
    "name": "string",
    "output": {
        "type": "string",
        "width": "string",
        "pagination": "string",
        "theme": "string"
    },
    "owner": "string",
    "schedule": {
        "type": "string",
        "at": {
            "by_day": [],
            "tz": "string",
            "hour": 0,
            "minute": 0
        }
    },
    "until": "string"
}
```

GET /reports/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the report.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "cc": [],
    "description": "string",
    "email_message": "string",
    "email_subject": "string",
    "enabled": true,
    "from": "string",
    "id": 0,
    "include_links": "string",
    "name": "string",
    "output": {},
    "owner": "string",
    "schedule": {},
    "until": "string"
}
```

GET /reports/{id}/download

Specify the following parameters.

id: ***Number***

The unique identifier for the report.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "cc": [],
    "description": "string",
    "email_message": "string",
```

```
    "email_subject": "string",
    "enabled": true,
    "from": "string",
    "id": 0,
    "include_links": "string",
    "name": "string",
    "output": {},
    "owner": "string",
    "schedule": {},
    "until": "string"
}
```

DELETE /reports/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

GET /reports/{id}/contents

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "dashboard_id": 0,
    "params": {},
    "type": "string"
}
```

PUT /reports/{id}/contents

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

body: ***Object***
The contents of the report.

POST /reports/{id}/emailgroups/{group-id}

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

group-id: ***Number***
The unique identifier for the email group.

POST /reports/{id}/emailgroups

Specify the following parameters.

id: ***Number***
The unique identifier for the report.

body: ***Object***

The list of email group IDs to assign or unassign to the report.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

GET /reports/{id}/emailgroups

Specify the following parameters.

id: ***Number***

The unique identifier for the report.

DELETE /reports/{id}/emailgroups/{group-id}

Specify the following parameters.

id: ***Number***

The unique identifier for the report.

group-id: ***Number***

The unique identifier for the email group.

Running config

The running configuration file is a JSON document that contains core system configuration information for the ExtraHop system.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /runningconfig	Retrieve the current running configuration file.
PUT /runningconfig	Replace the current running configuration file. Configuration file changes are not automatically saved.
POST /runningconfig/save	Save the current changes to the running configuration file.
GET /runningconfig/saved	Retrieve the saved running configuration file.

Operation details

GET /runningconfig/saved

There are no parameters for this operation.

POST /runningconfig/save

There are no parameters for this operation.

GET /runningconfig

Specify the following parameters.

section: ***String***
(Optional) (Optional) The specific section of the running configuration file that you want to retrieve.

PUT /runningconfig

Specify the following parameters.

body: ***String***
(Optional) The running configuration file.

Software

You can view a list of software that the ExtraHop system has observed on your network.

Operation	Description
GET /software	Retrieve software observed by the ExtraHop system.
GET /software/{id}	Retrieve software observed by the ExtraHop system by ID.

Operation details

GET /software

Specify the following parameters.

software_type: ***String***
(Optional) The type of software.
name: ***String***
(Optional) The name of the software.
version: ***String***
(Optional) The version of the software.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "id": "string",
    "name": "string",
    "software_type": "string",
    "version": "string"
}
```

GET /software/{id}

Specify the following parameters.

id: ***String***

The unique identifier for the software.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "id": "string",
    "name": "string",
    "software_type": "string",
    "version": "string"
}
```

SSL decrypt key

This resource enables you to add a decryption key for your network traffic.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /ssldecryptkeys	Retrieve all SSL decryption keys.
POST /ssldecryptkeys	Create a new SSL decryption key.
DELETE /ssldecryptkeys/{id}	Remove an SSL key from the ExtraHop system.
GET /ssldecryptkeys/{id}	Retrieve an SSL PEM and metadata.
PATCH /ssldecryptkeys/{id}	Update an existing SSL decryption key.
GET /ssldecryptkeys/{id}/protocols	Retrieve all protocols assigned to an SSL decryption key.
POST /ssldecryptkeys/{id}/protocols	Create a new protocol for an SSL decryption key.
DELETE /ssldecryptkeys/{id}/protocols/{protocol}	Delete a protocol from an SSL decryption key.

Operation details

GET /ssldecryptkeys

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "cert_pem": "string",
    "enabled": true,
    "id": "string",
    "name": "string"
}
```

POST /ssldecryptkeys

Specify the following parameters.

body: ***Object***

Set the specified property values on the new SSL decryption key.

enabled: ***Boolean***

Indicate whether this SSL decryption key is active.

name: ***String***

The friendly name for the SSL decryption key.

certificate: ***String***

The SSL certificate associated with this decryption key.

private_key: ***String***

The SSL private key that decrypts traffic.

Specify the body parameter in the following JSON format.

```
{
    "certificate": "string",
    "enabled": true,
    "name": "string",
    "private_key": "string"
}
```

PATCH /ssldecryptkeys/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property updates to the SSL decryption key.

id: ***String***

The hexadecimal representation of the SHA-1 hash of the SSL decryption key. The string must not include delimiters.

GET /ssldecryptkeys/{id}

Specify the following parameters.

id: ***String***

The hexadecimal representation of the SHA-1 hash of the SSL decryption key. The string must not include delimiters.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "cert_pem": "string",
    "enabled": true,
    "id": "string",
    "name": "string"
}
```

DELETE /ssldecryptkeys/{id}

Specify the following parameters.

id: ***String***

The hexadecimal representation of the SHA-1 hash of the SSL decryption key. The string must not include delimiters.

GET /ssldecryptkeys/{id}/protocols

Specify the following parameters.

id: ***String***

The hexadecimal representation of the SHA-1 hash of the SSL decryption key. The string must not include delimiters.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "port": 0,
    "protocol": "string"
}
```

POST /ssldecryptkeys/{id}/protocols

Specify the following parameters.

body: ***Object***

The body of the protocol.

protocol: ***String***

The name of the protocol, in lowercase.

port: ***Number***

The port in which to listen for traffic.

Specify the body parameter in the following JSON format.

```
{
    "port": 0,
    "protocol": "string"
}
```

id: ***String***

The unique identifier for the SSL decrypt key.

DELETE /ssldecryptkeys/{id}/protocols/{protocol}

Specify the following parameters.

protocol: ***String***

The name of the protocol, in lowercase.

id: ***String***

The hexadecimal representation of the SHA-1 hash of the SSL decryption key. The string must not include delimiters.

port: ***Number***

(Optional) Remove only the protocols that are assigned on this port.

Support pack

A support pack is a file that contains configuration adjustments provided by ExtraHop Support.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /supportpacks	Retrieve metadata about all support packs.
POST /supportpacks/execute	Run a new support pack.

Operation	Description
GET /supportpacks/queue/{id}	Check on the status of an in-progress, running support pack.
GET /supportpacks/{filename}	Downlolad an existing support pack by filename.

Operation details

GET /supportpacks/queue/{id}

Specify the following parameters.

id: ***String***

The unique identifier for the running support pack.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "created_time": 0,
    "filename": "string",
    "size": "string"
}
```

GET /supportpacks/{filename}

Specify the following parameters.

filename: ***String***

The name of the support pack to download.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "created_time": 0,
    "filename": "string",
    "size": "string"
}
```

POST /supportpacks/execute

GET /supportpacks

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "created_time": 0,
    "filename": "string",
    "size": "string"
}
```

Tag

Device tags enable you to associate a device or group of devices by some characteristic.

For example, you might tag all of your HTTP servers or tag all of the devices that are in a common subnet. For more information, see Tag a device through the REST API.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /tags	Retrieve all tags.
POST /tags	Create a a new tag.
DELETE /tags/{id}	Delete a specific tag.
GET /tags/{id}	Retrieve a specific tag.
PATCH /tags/{id}	Apply updates to a specific tag.
GET /tags/{id}/devices	Retrieve all devices that are assigned to a specific tag.
POST /tags/{id}/devices	Assign and unassign a specific tag to devices.
DELETE /tags/{id}/devices/{child-id}	Unassign a device from a specific tag.
POST /tags/{id}/devices/{child-id}	Assign a device to a specific tag.

Operation details

GET /tags

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "id": 0,
    "mod_time": 0,
    "name": "string"
}
```

POST /tags

Specify the following parameters.

body: ***Object***

Apply the specified property values to the new tag.

name: ***String***

The string value for the tag.

Specify the body parameter in the following JSON format.

```
{
    "name": "string"
}
```

GET /tags/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the tag.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "id": 0,
    "mod_time": 0,
    "name": "string"
}
```

DELETE /tags/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the tag.

PATCH /tags/{id}

Specify the following parameters.

body: ***Object***
Apply the specified property value updates to the tag.
id: ***Number***
The unique identifier for the tag.

GET /tags/{id}/devices

Specify the following parameters.

id: ***Number***
The unique identifier for the tag.

POST /tags/{id}/devices

Specify the following parameters.

body: ***Object***
Lists of unique identifies for device to assign and unassign.
assign: ***Array of Numbers***
IDs of resources to assign
unassign: ***Array of Numbers***
IDs of resources to unassign
Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***
The unique identifier for the tag.

POST /tags/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device.

id: ***Number***

the unique identifier for the tag.

DELETE /tags/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device.

id: ***Number***

The unique identifier for the tag.

Threat Collection

The Threat Collection resource enables you to upload free and commercial threat collections offered by the security community to your Reveal(x) system.

- You must upload threat collections individually to your Command appliance or Reveal(x) 360, and to all connected sensors.
- Custom threat collections must be formatted in Structured Threat Information eXpression (STIX) as TAR.GZ files. Reveal(x) currently supports STIX version 1.0 - 1.2.
- You can directly upload threat collections to Reveal(x) 360 systems for self-managed sensors. Contact ExtraHop Support to upload a threat collection to ExtraHop-managed sensors.
- The maximum number of observables that a threat collection can contain depends on your platform and license. Contact your ExtraHop representative for more information.

Note: This topic applies only to ExtraHop Reveal(x) Premium and Ultra.

For information about uploading STIX files through the ExtraHop system, see Upload STIX files through the REST API.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /threatcollections	Retrieve all threat collections.
POST /threatcollections	Create a new threat collection.
DELETE /threatcollections/{id}	Delete a threat collection.
PUT /threatcollections/{id}	Upload a new threat collection. ExtraHop currently supports STIX versions 1.0 - 1.2. **Note:** If a threat collection with the same name already exists on the ExtraHop system, the existing threat collection is overwritten.
GET /threatcollections/{id}/observables	Retrieve the number of STIX observables loaded from a threat collection, such as IP address, hostname, or URI.

Operation details

GET /threatcollections

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "id": 0,
    "last_updated": 0,
    "name": "string",
    "observables": 0,
    "user_key": "string"
}
```

POST /threatcollections

Specify the following parameters.

user_key: ***String***
: (Optional) The user-supplied identifier for the threat collection. If this parameter is not specified, the threat collection name is set for this value, without spaces or punctuation.

name: ***String***
: The name for the threat collection.

file: ***Filename***
: The filename for the threat collection.

PUT /threatcollections/~{userKey}

Specify the following parameters.

userKey: ***String***
: The user-supplied identifier for the threat collection.

name: ***String***
: (Optional) The name for the threat collection.

file: ***Filename***
: (Optional) The filename for the threat collection.

DELETE /threatcollections/{id}

Specify the following parameters.

id: ***String***
: The unique identifier for the threat collection.

GET /threatcollections/{id}/observables

Specify the following parameters.

id: ***String***
: The unique identifier for the threat collection.

Trigger

Triggers are custom scripts that perform an action upon a pre-defined event.

For example, you can write a trigger to record a custom metric every time an HTTP request occurs, or classify traffic for a particular server as an Application server. For more information, see the Trigger API Reference. For supplemental implementation notes about advanced options, see Advanced trigger options.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /triggers	Retrieve all triggers.
POST /triggers	Create a new trigger.
POST /triggers/externaldata	Sends data to the Trigger API by running the EXTERNAL_DATA event. You can access the data through the ExternalData trigger class. **Note:** This operation is not available for Command appliances or Reveal(x) 360.
DELETE /triggers/{id}	Delete a specific identifier.
GET /triggers/{id}	Retrieve a specific trigger by unique identifier.
PATCH /triggers/{id}	Update an existing trigger.
GET /triggers/{id}/devicegroups	Retrieve all device groups that are assigned to a specific trigger.
POST /triggers/{id}/devicegroups	Assign and unassign a specific trigger to device groups.
DELETE /triggers/{id}/devicegroups/{child-id}	Unassign a device group from a specific trigger.
POST /triggers/{id}/devicegroups/{child-id}	Assign a device group to a specific trigger.
GET /triggers/{id}/devices	Retrieve all devices that are assigned to a specific trigger.
POST /triggers/{id}/devices	Assign and unassign a specific trigger to devices.
DELETE /triggers/{id}/devices/{child-id}	Unassign a device from a specific trigger.
POST /triggers/{id}/devices/{child-id}	Assign a device to a specific trigger.

Operation details

GET /triggers

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "apply_all": true,
    "author": "string",
    "debug": true,
    "description": "string",
    "disabled": true,
    "event": "string",
```

```
    "events": [
        "string"
    ],
    "hints": {},
    "id": 0,
    "mod_time": 0,
    "name": "string",
    "script": "string"
}
```

DELETE /triggers/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the trigger.

POST /triggers/externaldata

Specify the following parameters.

body: ***Object***

The object containing the data to send to triggers through the EXTERNAL_DATA event.

type: ***String***

A string identifier that describes the data contained in the body parameter. For example, you could specify 'phantom-data' for data sent from the Phantom SOAR platform.

body: ***Object***

The data to send to triggers through the EXTERNAL_DATA event. This data can be accessed in the trigger with the 'ExternalData.body' property.

Specify the body parameter in the following JSON format.

```
{
    "body": {},
    "type": "string"
}
```

POST /triggers

Specify the following parameters.

body: ***Object***

The property values for the new trigger.

name: ***String***

The friendly name for the trigger.

description: ***String***

(Optional) An optional description of the trigger.

author: ***String***

The name of the creator of the trigger.

script: ***String***

The JavaScript content of the trigger.

event: ***String***

(Optional) Deprecated. Replaced by the events field.

events: ***Array of Strings***

The list of events on which the trigger runs, expressed as a JSON array.

disabled: ***Boolean***

Indicates whether the trigger can run.

debug: ***Boolean***

Indicates whether debug statements are printed for the trigger.

apply_all: ***Boolean***

Indicates whether the trigger applies to all relevant resources.

hints: ***Object***

Options that are based on selected trigger events. For more information about the hints object, see the REST API Guide.

Specify the body parameter in the following JSON format.

```
{
    "apply_all": true,
    "author": "string",
    "debug": true,
    "description": "string",
    "disabled": true,
    "event": "string",
    "events": [
        "string"
    ],
    "hints": {},
    "name": "string",
    "script": "string"
}
```

PATCH /triggers/{id}

Specify the following parameters.

body: ***Object***

The property value updates for the trigger.

id: ***Number***

The unique identifier for the trigger.

GET /triggers/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the trigger.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "apply_all": true,
    "author": "string",
    "debug": true,
    "description": "string",
    "disabled": true,
    "event": "string",
    "events": [
        "string"
    ],
```

```
    "hints": {},
    "id": 0,
    "mod_time": 0,
    "name": "string",
    "script": "string"
}
```

GET /triggers/{id}/devicegroups

Specify the following parameters.

id: ***Number***

The unique identifier for the trigger.

POST /triggers/{id}/devicegroups

Specify the following parameters.

body: ***Object***

A list of unique identifiers for device groups that are assigned and unassigned to a trigger.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the trigger.

POST /triggers/{id}/devicegroups/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device group.

id: ***Number***

The unique identifier for the trigger.

DELETE /triggers/{id}/devicegroups/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device group.

id: ***Number***

The unique identifier for the trigger.

GET /triggers/{id}/devices

Specify the following parameters.

id: ***Number***

The unique identifier for the trigger.

POST /triggers/{id}/devices

Specify the following parameters.

body: ***Object***

A list of unique identifiers for devices that are assigned and unassigned to a trigger.

assign: ***Array of Numbers***

IDs of resources to assign

unassign: ***Array of Numbers***

IDs of resources to unassign

Specify the body parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

id: ***Number***

The unique identifier for the trigger.

POST /triggers/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device.

id: ***Number***

The unique identifier for the trigger.

DELETE /triggers/{id}/devices/{child-id}

Specify the following parameters.

child-id: ***Number***

The unique identifier for the device.

id: ***Number***

The unique identifier for the trigger.

Advanced trigger options

Advanced trigger options are configuration options that you can set depending on the system events associated with the trigger. For example, you can configure the number of payload bytes to buffer on HTTP request events.

Advanced options are contained in the `hints` object of the trigger resource as shown in the following example:

```
"hints": {
"flowClientPortMin": null,
"flowClientBytes": 16384,
"flowClientPortMax": null,
"flowServerBytes": 16384,
"flowPayloadTurn": true,
```

```
"flowServerPortMin": 135,
"flowServerPortMax": 49155
}
```

The following table describes available advanced options and applicable events:

Option	Description	Applicable events
`"snaplen": number`	Specifies the number of bytes to capture per packet, up to a maximum of 65535. The capture starts with the first byte in the packet. Specify this option only if the trigger script captures packets. A value of `0` configures the trigger to capture the maximum number of bytes for each packet.	All events except: • ALERT_RECORD_COMMIT • METRIC_CYCLE_BEGIN • METRIC_CYCLE_END • FLOW_REPORT • NEW_APPLICATION • NEW_DEVICE • SESSION_EXPIRE
`"payloadBytes": number`	Specifies the minimum number of payload bytes to buffer.	• CIFS_REQUEST • CIFS_RESPONSE • HTTP_REQUEST • HTTP_RESPONSE • ICA_TICK
`"clipboardBytes": number`	Specifies the number of bytes to buffer on a Citrix clipboard transfer.	• ICA_TICK
`"cycle": [30sec, 5min, 1hr, 24hr]`	Specifies the length of the metric cycle, expressed in seconds.	• METRIC_CYCLE_BEGIN • METRIC_CYCLE_END • METRIC_RECORD_COMMIT
`"metricTypes": string`	Specifies the metric type by the raw metric name such as extrahop.device.http_server.	• ALERT_RECORD_COMMIT • METRIC_RECORD_COMMIT
`"flowPayloadTurn": boolean`	Enables packet capture on each flow turn. Per-turn analysis continuously analyzes communication between two endpoints to extract a single payload data point from the flow. If this option is enabled, any values specified for the `flowClientString` and `flowServerString` options are ignored.	• SSL_PAYLOAD • TCP_PAYLOAD
`"flowClientPortMin": number`	Specifies the minimum port number of the client port range. Valid values are `0` to `65535`. A value of `0` specifies matching of any port.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD

Option	Description	Applicable events
`"flowClientPortMax": number`	Specifies the maximum port number of the client port range. Valid values are `0` to `65535`. Any value specified for this option is ignored if the value of the `flowClientPortMin` option is `0`.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
`"flowClientBytes": number`	Specifies the number of client bytes to buffer. The value of this option cannot be set to `0` if the value of the `flowServerBytes` option is also set to `0`.	• SSL_PAYLOAD • TCP_PAYLOAD
`"flowClientString": string`	Specifies the format string of client data to process. Any value specified for this option is ignored if the `flowPayloadTurn` option is enabled.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
`"flowServerPortMin": number`	Specifies the minimum port number of the server port range. Valid values are `0` to 65535. A value of `0` specifies matching of any port.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
`"flowServerPortMax": number`	Specifies the maximum port number of the server port range. Valid values are `0` to `65535`. Any value specified for this option is ignored if the value of the `flowServerPortMin` option is `0`.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
`"flowServerBytes": number`	Specifies the number of server bytes to buffer. The value of this option cannot be set to `0` if the value of the `flowClientBytes` option is also set to `0`.	• SSL_PAYLOAD • TCP_PAYLOAD
`"flowServerString": string`	Specifies the format string of server data to process. Returns the entire packet upon a string match. Any value specified for this option is ignored if the	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD

Option	Description	Applicable events
	`flowPayloadTurn` option is enabled.	
`"flowUdpAll": boolean`	Enables capture of all UDP datagrams.	• UDP_PAYLOAD
`"fireClassifyOnExpiration": boolean`	Enables running the event upon expiration in order to accumulate metrics for flows that were not classified before expiring.	• FLOW_CLASSIFY

User

The user resource enables you to create and manage the list of users who have access to the ExtraHop system and the privilege levels for those users.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /users	Retrieve all users.
POST /users	Create a new user.
DELETE /users/{username}	Delete a specific user.
GET /users/{username}	Retrieve a specific user.
PATCH /users/{username}	Update settings for a specific user.
GET /users/{username}/apikeys	Retrieve all API keys for a specific user.
GET /users/{username}/apikeys/{keyid}	Retrieve information about a specific API key and user.

Operation details

GET /users

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "date_joined": "string",
    "effective_roles": {},
    "eh_account_team": true,
    "enabled": true,
    "granted_roles": {},
    "last_ui_login_time": "string",
    "name": "string",
    "type": "string",
    "username": "string"
}
```

POST /users

Specify the following parameters.

body: ***Object***

The user account settings.

enabled: ***Boolean***

(Optional) Indicates whether the user can login to the ExtraHop system.

name: ***String***

The friendly name for the user.

username: ***String***

The login name for the user.

password: ***String***

The password for the user. Passwords must meet the requirements configured in the Administration settings.

granted_roles: ***Object***

(Optional) The privileges for the user. Supported permission levels are described in the REST API Guide.

create_apikey: ***Boolean***

(Optional) Generate and return a new API key for the created user.

type: ***String***

(Optional) The authentication method used by this user to log in.

The following values are valid:

- `local`
- `remote`

eh_account_team: ***Boolean***

Indicates an ExtraHop Account Team user that accesses the ExtraHop system through ExtraHop Cloud Services.

Specify the body parameter in the following JSON format.

```
{
    "create_apikey": true,
    "eh_account_team": true,
    "enabled": true,
    "granted_roles": {},
    "name": "string",
    "password": "string",
    "type": "string",
    "username": "string"
}
```

GET /users/{username}

Specify the following parameters.

username: ***String***

The name of the user.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "date_joined": "string",
    "effective_roles": {},
    "eh_account_team": true,
    "enabled": true,
    "granted_roles": {},
```

```
    "last_ui_login_time": "string",
    "name": "string",
    "type": "string",
    "username": "string"
}
```

PATCH /users/{username}

Specify the following parameters.

body: ***Object***

The user account settings.

enabled: ***Boolean***

(Optional) Indicates whether the user can login to the ExtraHop system.

name: ***String***

(Optional) The friendly name for the user.

password: ***String***

(Optional) The password for the user. Passwords must meet the requirements configured in the Administration settings.

granted_roles: ***Object***

(Optional) The privileges for the user. Supported permission levels are described in the REST API Guide.

Specify the body parameter in the following JSON format.

```
{
    "enabled": true,
    "granted_roles": {},
    "name": "string",
    "password": "string"
}
```

username: ***String***

The name of the user.

DELETE /users/{username}

Specify the following parameters.

username: ***String***

The name of the user.

dest_user: ***String***

(Optional) The user that customizations are transferred to. If this parameter is specified, all dashboards, collections, and activity maps owned by the deleted user are transferred to this user.

GET /users/{username}/apikeys

Specify the following parameters.

username: ***String***

The name of the user.

GET /users/{username}/apikeys/{keyid}

Specify the following parameters.

keyid: ***String***

The ID of the API key.

username: ***String***

The name of the user.

User group

The user group resource enables you to manage and update groups of users and their dashboard sharing associations.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /usergroups	Retrieve all user groups.
POST /usergroups	Create a new user group.
POST /usergroups/refresh	Query LDAP for the most recent user memberships for all remote user groups.
DELETE /usergroups/{id}	Delete a specific user group.
GET /usergroups/{id}	Retrieve a specific user group.
PATCH /usergroups/{id}	Update a specific user group.
DELETE /usergroups/{id}/associations	Delete all dashboard sharing associations with a specific user group.
GET /usergroups/{id}/members	Retrieve all members of a specific user group.
PATCH /usergroups/{id}/members	Assign or unassign users from a user group.
PUT /usergroups/{id}/members	Replace user group assignments.
POST /usergroups/{id}/refresh	Query LDAP for the most recent user membership of a specific remote user group.

Operation details

GET /usergroups

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "display_name": "string",
    "enabled": true,
    "id": "string",
    "is_remote": true,
    "last_sync_time": 0,
    "name": "string",
    "rights": []
}
```

POST /usergroups

Specify the following parameters.

body: ***Object***

The properties of the user group.

name: ***String***

The name for the user group.

enabled: ***Boolean***

Indicates whether the user group is enabled.

Specify the body parameter in the following JSON format.

```
{
    "enabled": true,
    "name": "string"
}
```

POST /usergroups/refresh

There are no parameters for this operation.

PATCH /usergroups/{id}

Specify the following parameters.

body: ***Object***

The property value updates for the specific user group.

id: ***String***

The unique identifier for the user group.

GET /usergroups/{id}

Specify the following parameters.

id: ***String***

The unique identifier for the user group.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "display_name": "string",
    "enabled": true,
    "id": "string",
    "is_remote": true,
    "last_sync_time": 0,
    "name": "string",
    "rights": []
}
```

DELETE /usergroups/{id}

Specify the following parameters.

id: ***String***

The unique identifier for the user group.

DELETE /usergroups/{id}/associations

Specify the following parameters.

id: ***String***
The unique identifier for the user group.

POST /usergroups/{id}/refresh

Specify the following parameters.

id: ***String***
The unique identifier for the user group.

GET /usergroups/{id}/members

Specify the following parameters.

id: ***String***
The unique identifier for the user group.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "users": {}
}
```

PATCH /usergroups/{id}/members

Specify the following parameters.

id: ***String***
The unique identifier for the user group.

body: ***String***
An object that specifies which users to asssign or unassign. Each key must be a username and each value must be either "member" or null. For example {"Alice": "member", "Bob": null} assigns Alice to the group and unassigns Bob from the group.

PUT /usergroups/{id}/members

Specify the following parameters.

id: ***String***
The unique identifier for the user group.

body: ***String***
An object that specifies which users are assigned to the group. Each key must be a username and each value must be "member". For example {"Alice": "member", "Bob": "member"} assigns Alice and Bob as the only members of the group.

VLAN

Virtual LANs are logical groupings of traffic or devices on the network.

The following table displays all of the operations you can perform on this resource:

Operation	Description
GET /vlans	Retrieve all VLANs
GET /vlans/{id}	Retrieve a specific VLAN.

Operation	Description
PATCH /vlans/{id}	Update a specific VLAN.

Operation details

GET /vlans

There are no parameters for this operation.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string",
    "network_id": 0,
    "node_id": 0,
    "vlanid": 0
}
```

GET /vlans/{id}

Specify the following parameters.

id: ***Number***

The unique identifier for the VLAN.

If the request is successful, the ExtraHop system returns an object in the following format.

```
{
    "description": "string",
    "id": 0,
    "mod_time": 0,
    "name": "string",
    "network_id": 0,
    "node_id": 0,
    "vlanid": 0
}
```

PATCH /vlans/{id}

Specify the following parameters.

body: ***Object***

Apply the specified property value updates to the VLAN.

id: ***Number***

The unique identifier for the VLAN.

Watchlist

To guarantee that an asset, such as an important server, database, or laptop, is guaranteed Advanced Analysis, you can add that device to the watchlist.

Tip: If you want to add several devices to the watchlist, consider creating a device group and then prioritizing that group for Advanced Analysis.

Here are important considerations about the watchlist:

- The watchlist only applies to Advanced Analysis.
- The watchlist can contain as many devices as allowed by the Advanced Analysis capacity, which is determined by your license.
- A device stays on the watchlist whether it is inactive or active. A device has to be active for the ExtraHop system to collect Advanced Analysis metrics.

For more information about Advanced Analysis, see Analysis levels.

The following table displays all of the operations you can perform on this resource:

Operation	Description
DELETE /watchlist/device/{id}	Remove a device from the watchlist.
POST /watchlist/device/{id}	Add a device to the watchlist.
GET /watchlist/devices	Retrieve all devices that are in the watchlist.
POST /watchlist/devices	Add or remove devices from the watchlist.

Operation details

GET /watchlist/devices

There are no parameters for this operation.

POST /watchlist/device/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the device.

DELETE /watchlist/device/{id}

Specify the following parameters.

id: ***Number***
The unique identifier for the device.

POST /watchlist/devices

Specify the following parameters.

assignments: ***Object***
A list of devices to add to or remove from the watchlist.

assign: ***Array of Numbers***
IDs of resources to assign

unassign: ***Array of Numbers***
IDs of resources to unassign

Specify the assignments parameter in the following JSON format.

```
{
    "assign": [],
    "unassign": []
}
```

ExtraHop REST API examples

The following examples demonstrate common REST API operations.

- Change a dashboard owner through the REST API
- Extract the device list through the REST API
- Create and assign a device tag through the REST API
- Query for metrics about a specific device through the REST API
- Create, retrieve, and delete an object through the REST API
- Query the record log

Upgrade ExtraHop firmware through the REST API

You can automate upgrades to the firmware on your ExtraHop appliances through the ExtraHop REST API. This guide provides instructions to upgrade through the REST API Explorer, a cURL command, and a Python script.

Note: If your appliance is connected to ExtraHop Cloud Services, you can simplify the upgrade process by viewing available firmware versions and downloading firmware directly to the system from ExtraHop Cloud Services. For more information, see Upgrade ExtraHop firmware through the REST API with ExtraHop Cloud Services.

While the firmware upgrade process is similar across all ExtraHop appliances, some appliances have additional considerations or steps that you must address before you install the firmware in your environment. If you need assistance with your upgrade, contact ExtraHop Support.

All appliances must meet the following requirements:

- The firmware version must be compatible with your appliance model.
- The firmware version on your appliance must be supported by the upgrade version.
- Command appliances must be running firmware that is greater than or equal to their connected appliances.
- Discover appliances must be running firmware that is greater than or equal to connected Explore and Trace appliances.

If your deployment only includes a sensor, proceed to the API Explorer, cURL or Python upgrade instructions.

If your deployment includes additional appliance types, you must address the following dependencies before proceeding with the upgrade instructions.

If your deployment includes...	Pre-upgrade tasks	Upgrade order
Command appliances	Reserve a maintenance window of an hour for Command appliances managing 50,000 devices or more.	• Command appliance • Discover appliances • All Explore appliances (manager nodes, then data nodes) • Trace appliances
Explore appliances	See Upgrading ExtraHop recordstores.	
Trace appliances	None	

Upgrade ExtraHop firmware through the REST API Explorer

Important: The REST API Explorer is not available on Reveal(x) 360.

Download firmware and upgrade the appliance

1. Click **POST /extrahop/firmware/download/url**.
2. Click **Try it out**.
3. In the body field, specify the following fields:

   • **firmware_url:** The URL that the firmware .tar file can be downloaded from.

   • **upgrade:** Indicates whether to upgrade the appliance after the firmware download completes. Set this field to `true`.

   The body field should look similar to the following example text:

   ```
   {
     "upgrade": true,
     "firmware_url": "https://example.extrahop.com/eda/8.7.1.tar"
   }
   ```

4. Click **Send Request**.

   In the Response headers, note the value after the last forward slash in the `location` header. You will need this value to monitor the progress of the upgrade job. For example, the job ID in the following example is `ebbdbc9e-7113-448c-ab9b-cc0ec2307702`

   ```
   /api/v1/jobs/ebbdbc9e-7113-448c-ab9b-cc0ec2307702
   ```

Monitor the progress of the upgrade job

1. Click **Jobs**.
2. Click **GET /jobs/{id}**.
3. In the id field, type the value you copied from the `location` header in the previous task.
4. Click **Send Request**.
5. In the Response body, view information about the job.

   The `status` field is `DONE` when the job is complete.

Upgrade ExtraHop firmware with cURL

You can upgrade the firmware on an appliance through the cURL command.

Before you begin

- The cURL tool must be installed on your machine.
- The system firmware .tar file must be downloaded on your machine.

1. Open a terminal application.
2. Download firmware and upgrade the appliance.

   Run the following command, where `YOUR_KEY` is the API key for your user account, `HOSTNAME` is the hostname of your ExtraHop appliance, and `FIRMWARE_URL` is the URL that the firmware .tar file can be downloaded from:

   ```
   curl -v -X POST https://HOSTNAME/api/v1/extrahop/firmware/download/url -H
    "Authorization: ExtraHop apikey=YOUR_KEY" -H "Content-Type: application/
   json" -d "{ \"upgrade\": true, \"firmware_url\": \"FIRMWARE_URL\"}"
   ```

   In the command output, note the job ID in the Location header. For example, the job ID in the following example is `ebbdbc9e-7113-448c-ab9b-cc0ec2307702`:

   ```
   < Location: /api/v1/jobs/ebbdbc9e-7113-448c-ab9b-cc0ec2307702
   ```

3. Monitor the progress of the upgrade job.

Run the following command, where YOUR_KEY is the API key for your user account HOSTNAME is the hostname of your appliance, and JOB_ID is the ID you recorded in the previous step:

```
curl -v -X GET https://HOSTNAME/api/v1/jobs/JOB_ID -H "Authorization:
 ExtraHop apikey=API_KEY"
```

The command displays an object that contains information about the upgrade job. The upgrade is complete when the status field is DONE. If the upgrade is not complete, wait a few minutes and run the command again.

Retrieve and run the example Python script

The ExtraHop GitHub repository contains an example Python script that upgrades multiple appliances by reading URLs, API keys, and firmware file paths from a CSV file.

Important: The example python script authenticates to the sensor or console through an API key, which is not compatible with the Reveal(x) 360 REST API. To run this script with Reveal(x) 360, you must modify the script to authenticate with API tokens. See the py_rx360_auth.py script in the ExtraHop GitHub repository for an example of how to authenticate with API tokens.

Note: The script does not automatically disable record ingest for ExtraHop recordstores. You must manually disable record ingest before running the script for an ExtraHop recordstores.

1. Go to the ExtraHop code-examples GitHub repository and download the contents of the upgrade_system directory to your local machine.
2. In a text editor, open the systems.csv file and replace the example values with the hostnames and API keys of your appliances.
3. Run the upgrade_system_url.py script.

   The following arguments are optional:

   --max-threads {int}

   Specifies the maximum number of concurrent threads. The default value is 2.

   --wait {float}

   Specifies the number of minutes to wait before checking the progress of an upgrade job. The default value is 0.5.

   For example, the following command upgrades a maximum of 3 appliances at a time:

   ```
   python3 upgrade_system_url.py --max-threads 3
   ```

   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)
   ```

Upgrading ExtraHop recordstores

Pre-upgrade tasks

Before upgrading an ExtraHop recordstore, you must halt record ingest. You can halt record ingest for all of the nodes in a cluster from a single node.

Note: The message Could not determine ingest status on some nodes and Error might appear on the Cluster Data Management page in the Administration settings of the

upgraded nodes until all nodes in the cluster are upgraded. These errors are expected and can be ignored.

1. Open a terminal application.
2. Run the following command, where `YOUR_KEY` is the API for your user account, and `HOSTNAME` is the hostname of your ExtraHop recordstore:

```
curl -X PATCH "https://HOST/api/v1/extrahop/cluster" -H "accept:
 application/json" -H "Authorization: ExtraHop apikey=YOUR_KEY" -H
 "Content-Type: application/json" -d "{ \"ingest_enabled\": false}"
```

Post-upgrade tasks

After you have upgraded all of the nodes in the recordstore cluster, enable record ingest.

1. Open a terminal application.
2. Run the following command, where `YOUR_KEY` is the API for your user account, and `HOSTNAME` is the hostname of your ExtraHop recordstore:

```
curl -X PATCH "https://HOST/api/v1/extrahop/cluster" -H "accept:
 application/json" -H "Authorization: ExtraHop apikey=YOUR_KEY" -H
 "Content-Type: application/json" -d "{ \"ingest_enabled\": false}"
```

Change a dashboard owner through the REST API

Dashboards are owned by the logged in user that created them. If a user is no longer with your company, you might need to change the owner of the dashboard to maintain that dashboard.

To transfer ownership of a dashboard, you need the dashboard ID and the username of the dashboard owner. You can only view the username of the owner of a dashboard through the REST API.

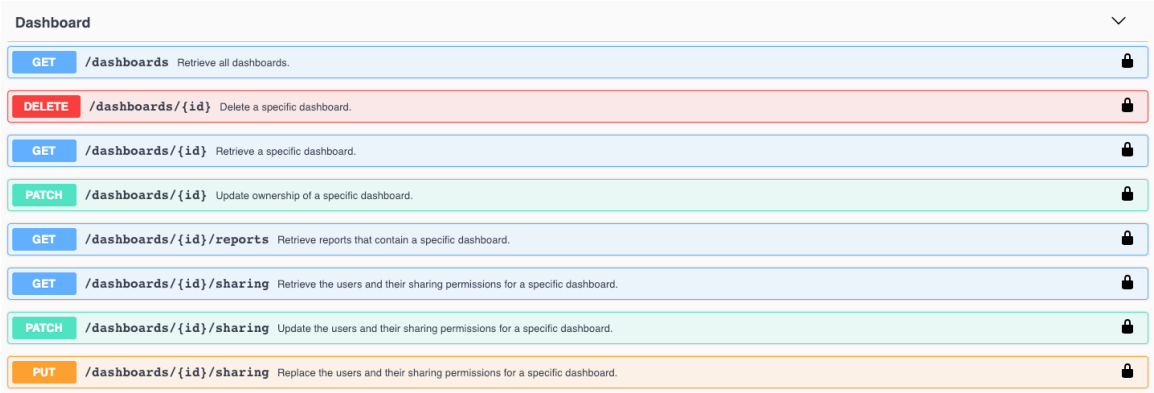
Before you begin

- You must log in to the sensor or console 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.

Retrieve the dashboard IDs

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 **Dashboard** to display dashboard operations.



5. Click **GET /dashboards**.
6. Click **Try it out** and then click **Send Request** to send the request to your sensor or console.
7. Search for the dashboards by the dashboard name or by the user account listed in the `"owner"` field. If your list of dashboards is long, you can press control-F and search the response body.
   For our example, we want to change the `"LDAP Server Health"` dashboard created by the user account for `"marksmith"`:

```
{
    "id": 1876,
    "comment": null,
    "mod_time": 1507576983922,
    "author": "Mark Smith",
    "name": "LDAP Server Health",
    "owner": "marksmith",
    "built-in": false,
    "short_code": "MpXgk",
    "rights": [
      "transfer",
      "view",
      "edit",
      "share",
      "delete"
    ]
}
```

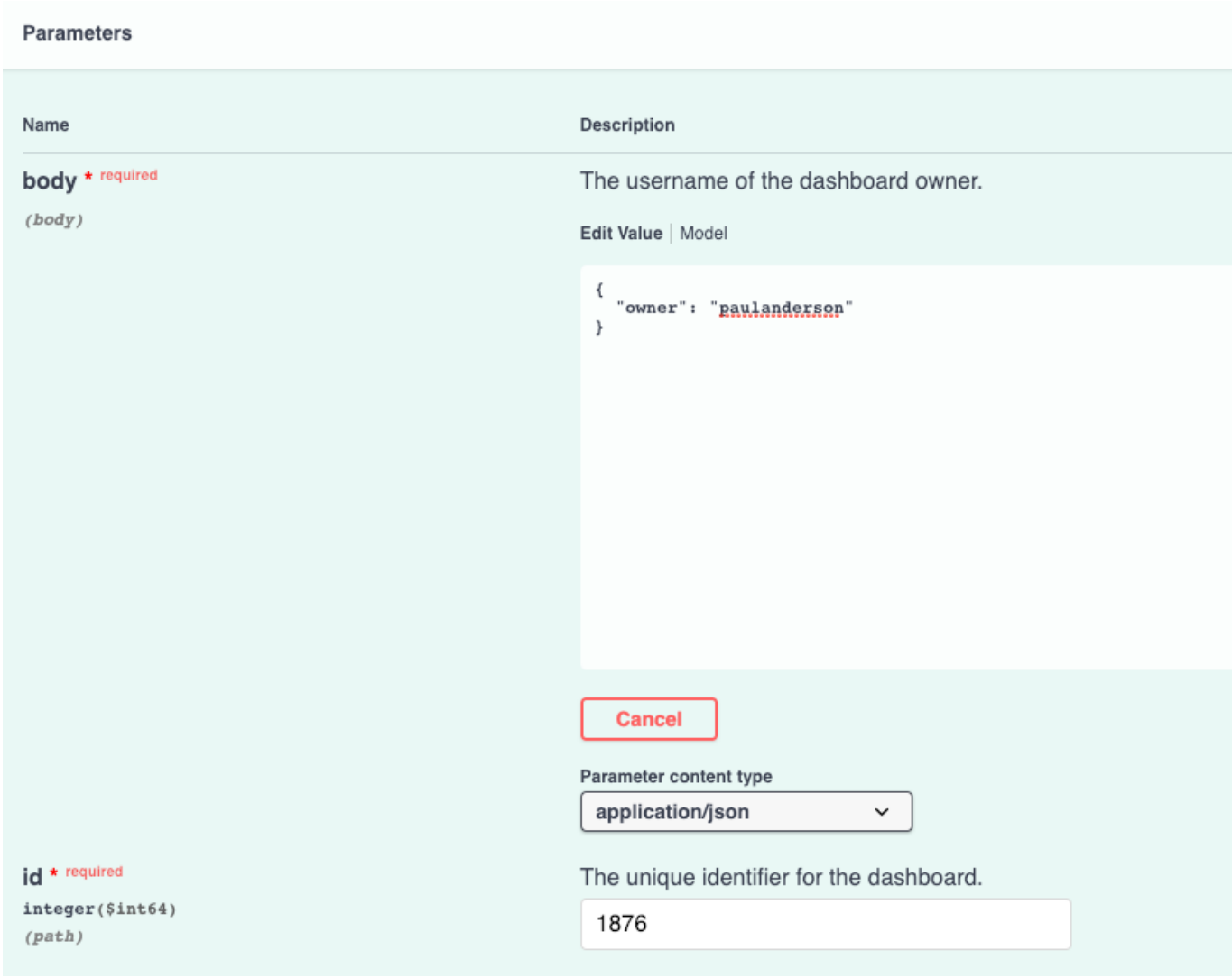
8. Note the number in the `"id"` field for each dashboard you want to modify.

Change the dashboard owner

1. Scroll down the page of Dashboard operations to the /dashboards/{id} section.
2. Click **PATCH /dashboards/{id}**.
3. Click **Try it out**.
   The JSON schema is automatically added to the body parameter text box.
4. In the body text box, in the `"owner"` field, replace `string` with the username of the new owner.
5. In the **id** field, type the number you previously noted for the dashboard.

   For our example, this value is `1876`. (You can only modify one dashboard at a time through the REST API Explorer.)

   In the following figure, we added the JSON `"string"` for the `"owner"` parameter to the body parameter text box, changed `"string"` to `"paulanderson"`, and typed `"1876"` in the id field.



6. Click **Send Request** to send the request to your sensor or console.
Under Server response, the Code column displays `204` if the operation is successful. You can click **GET /dashboards** again to verify that the `"owner"` field has changed. Note that you can only change the dashboard owner. You cannot change the dashboard name or author fields through the REST API.

   The dashboard is now available under **My Dashboards** in the ExtraHop system for the new user. As the new owner, you can now log in to your ExtraHop system and change other dashboard properties, such as the dashboard name or author.

Tip: After you click **Send Request**, the REST API Explorer provides scripts for the operation in Curl, Python 2.7, or Ruby.

Python script example

The ExtraHop GitHub repository contains an example Python script that searches for all dashboards owned by a user account on a sensor or console and then changes the owner for all of those dashboards to another user account.

Important: The example python script authenticates to the sensor or console through an API key, which is not compatible with the Reveal(x) 360 REST API. To run this script with Reveal(x) 360, you must modify the script to authenticate with API tokens. See the py_rx360_auth.py script in the ExtraHop GitHub repository for an example of how to authenticate with API tokens.

1. Go to the ExtraHop code-examples GitHub repository and download the `change_dashboard_owner/change_dashboard_owner.py` file to your local machine.
2. In a text editor, open the `change_dashboard_owner.py` file and replace the following configuration variables with information from your environment:
   - **HOST:** The IP address or hostname of the sensor or console.
   - **API_KEY:** The API key.
   - **CURRENT:** The username of the current dashboard owner.
   - **NEW:** The username of the new dashboard owner.
3. Run the following command:

```
python3 change_dashboard_owner.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)
```

Extract the device list through the REST API

The ExtraHop REST API enables you to extract the list of devices discovered by the sensor or console. By extracting the list with a REST API script, you can export the list in a format that can be read by third-party applications, such as a configuration management database (CMDB). In this topic, we show methods for extracting a list through both the cURL command 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.)

Retrieve the device list with the cURL command

The device list includes all device metadata, such as MAC addresses and device IDs. However, you can filter the list of devices with a JSON parser to extract the specific information you want to export. In this example, the device list is retrieved and then filtered with the jq parser to only extract the display name of each device.

Note: The following procedure is not compatible with the Reveal(x) 360 REST API. To retrieve the device list from Reveal(x) 360, see Retrieve the device list from Reveal(x) 360 with the cURL command.

Before you begin

- The cURL tool must be installed on your machine.
- The jq parser must be installed on your machine. For more information, see https://stedolan.github.io/jq/.

Open a terminal application and run the following command, where YOUR_KEY is the API for your user account, HOSTNAME is the hostname of your sensor or console, and MAX_DEVICES is a number large enough to be more than the total number of devices discovered by your system:

```
curl -s -X GET --header "Accept: application/json" --header
 "Authorization: ExtraHop apikey=YOUR_KEY"  "https://HOSTNAME/api/v1/
devices?active_from=1&active_until=0&limit=MAX_DEVICES" | jq -r '.[]
 | .display_name'
```

Note: If the command returns no results, make sure that a trusted certificate has been added to your ExtraHop system. Alternatively, you can add the `--insecure` option to retrieve the device list from an ExtraHop system without a trusted certificate; however, this method is not secure and not recommended.

Tip: You can append the `select(.analysis == "LEVEL")` option to filter results by analysis level. For example, the following command limits the results to include only devices that are selected for advanced analysis:

```
curl -s -X GET --header "Accept: application/json" --header
 "Authorization: ExtraHop apikey=YOUR_KEY"  "https://HOSTNAME/
api/v1/devices?active_from=1&active_until=0&limit=10000000000"
 | jq -r '.[] | select(.analysis == "advanced") | .display_name'
```

Tip: You can append the `select(.critical == BOOLEAN)` option to filter results by the critical field. For example, the following command limits the results to include only devices that are identified as critical by the ExtraHop system:

```
curl -s -X GET --header "Accept: application/json" --header
 "Authorization: ExtraHop apikey=YOUR_KEY"  "https://HOSTNAME/
api/v1/devices?active_from=1&active_until=0&limit=10000000000"
 | jq -r '.[] | select(.critical == true) | .display_name'
```

Tip: You can append the `select(.cloud_instance_name != null)` option to filter results by the cloud instance name field. For example, the following command limits the results to include only devices with a cloud instance name:

```
curl -s -X GET --header "Accept: application/json" --header
 "Authorization: ExtraHop apikey=YOUR_KEY"  "https://HOSTNAME/
api/v1/devices?active_from=1&active_until=0&limit=10000000000"
  | jq -r '.[] | select(.cloud_instance_name != null)
 | .cloud_instance_name'
```

Retrieve the device list from Reveal(x) 360 with the cURL command

The device list includes all device metadata, such as MAC addresses and device IDs. However, you can filter the list of devices with a JSON parser to extract the specific information you want to export. In this example, the device list is retrieved and then filtered with the jq parser to only extract the display name of each device.

Note: The following procedure is only compatible with the Reveal(x) 360 REST API. To retrieve the device list from sensors and ECA VMs, see Retrieve the device list with the cURL command.

Before you begin

- The cURL tool must be installed on your machine.
- The jq parser must be installed on your machine. For more information, see https://stedolan.github.io/jq/.

1. Open a terminal application and run the following command, where REVEAL_X_360_REST_API is the hostname of the Reveal(x) 360 API. This hostname is displayed in Reveal(x) 360 on the API Access page under API Endpoint. The hostname does not include the /oauth/token:

```
HOST="https://REVEAL_X_360_REST_API"
```

2. Run the following command, where YOUR_ID is the ID of the REST API credentials:

```
ID="YOUR_ID"
```

3. Run the following command, where YOUR_SECRET is the secret of the REST API credentials:

```
SECRET="YOUR_SECRET"
```

4. Run the following command:

```
AUTH=$(printf "$ID:$SECRET" | base64 --wrap=0)
```

5. Run the following command:

```
ACCESS_TOKEN=$(curl -s \
    -H "Authorization: Basic ${AUTH}" \
    -H "Content-Type: application/x-www-form-urlencoded" \
    --request POST \
     ${HOST}/oauth2/token \
    -d "grant_type=client_credentials" \
    | jq -r '.access_token')
```

6. Run the following command, where MAX_DEVICES is a number large enough to be more than the total number of devices discovered by your system:

```
curl -s -X GET -H "Authorization: Bearer ${ACCESS_TOKEN}"  "$HOST/api/
v1/devices?active_from=1&active_until=0&limit=MAX_DEVICES" | jq -r '.[]
 | .display_name'
```

Tip: You can append the `select(.analysis == "LEVEL")` option to filter results by analysis level. For example, the following command limits the results to include only devices that are selected for advanced analysis:

```
curl -s -X GET -H "Authorization: Bearer
 ${ACCESS_TOKEN}"  "$HOST/api/v1/devices?
active_from=1&active_until=0&limit=10000000000" | jq -r '.[] |
 select(.analysis == "advanced") | .display_name'
```

Tip: You can append the `select(.critical == BOOLEAN)` option to filter results by the critical field. For example, the following command limits the results to include only devices that are identified as critical by the ExtraHop system:

```
curl -s -X GET -H "Authorization: Bearer
 ${ACCESS_TOKEN}"  "$HOST/api/v1/devices?
active_from=1&active_until=0&limit=10000000000"  | jq -r '.[] |
 select(.critical == true) | .display_name'
```

Tip: You can append the `select(.cloud_instance_name != null)` option to filter results by the cloud instance name field. For example, the following command limits the results to include only devices with a cloud instance name:

```
curl -s -X GET -H "Authorization: Bearer
 ${ACCESS_TOKEN}"  "$HOST/api/v1/devices?
```

```
active_from=1&active_until=0&limit=10000000000"  | jq -r '.[] |
 select(.cloud_instance_name != null) | .cloud_instance_name'
```

Retrieve and run the example Python script

The ExtraHop GitHub repository contains an example Python script that extracts the device list, including all device metadata, and writes the list to a CSV file in the same directory as the script.

1. Go to the ExtraHop code-examples GitHub repository and download the `extract_device_list/extract_device_list.py` file to your local machine.
2. In a text editor, open the `extract_device_list.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.
     - **FILENAME:** The file that output will be written to
     - **LIMIT:** The maximum number of devices to retrieve with each GET request
     - **SAVEL2:** Retrieves L2 parent devices. This variable is valid only if you have enabled the ExtraHop system to discover devices by IP address.
     - **ADVANCED_ONLY:** Retrieves only devices that are currently under advanced analysis
     - **HIGH_VALUE_ONLY:** Retrieves only devices that are considered high value
   - 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 `/oauth/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.
     - **FILENAME:** The file that output will be written to
     - **LIMIT:** The maximum number of devices to retrieve with each GET request
     - **SAVEL2:** Retrieves L2 parent devices. This variable is valid only if you have enabled the ExtraHop system to discover devices by IP address.
     - **ADVANCED_ONLY:** Retrieves only devices that are currently under advanced analysis
     - **HIGH_VALUE_ONLY:** Retrieves only devices that are considered high value
3. Run the following command:

```
python3 extract_device_list.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)
```

Create a trusted SSL certificate through the REST API

By default, sensors and consoles include a self-signed SSL certificate. However, you can improve the security and performance of your system by adding a trusted certificate signed by a certificate authority (CA). You can create the certificate signing request to send to your CA through the ExtraHop REST API. After you receive the signed certificate, you can also add it to your sensor or console through the REST API.

Before you begin

- You must log in to the sensor or console 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.

Note: You can also perform the procedures in this topic through the Administration settings. For more information, see the following topics:

- Create a certificate signing request from your ExtraHop system
- SSL Certificate

Create an SSL certificate signing request

To create a signed SSL certificate, you must send a certificate signing request to a trusted CA.

Important: The REST API Explorer is not available on Reveal(x) 360.

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 **ExtraHop** and then click **POST/extrahop/sslcert/signingrequest**.
5. Click **Try it out**.
   The JSON schema is automatically added to the SSL Certificate Signing Request Parameters parameter text box.
6. In the SSL Certificate Signing Request Parameters parameter text box, specify the certificate signing request fields.
   a) In the `common_name` field, replace `string` with the fully qualified domain name of your sensor or console.
   b) In the `subject_alternative_names` field, add one or more alternative domain names or IP addresses for your sensor or console.

      Note: The `subject_alternative_names` field is required. If your system has only one domain name, duplicate the value from the `common_name` field. You must include at least one subject alternative name with the type set to `dns`, but additional alternative names can have the type set to `ip` or `dns`.

   c) (Optional) In the `email_address` field, replace `string` with the email address of the certificate owner.
   d) (Optional) In the `organization_name` field, replace `string` with the registered legal name of your organization.
   e) (Optional) In the `country_code` field, replace `string` with the 2-character ISO country code of the country that your organization is located in.

f) (Optional) In the `state_or_province_name` field, replace `string` with the name of the state or that your organization is located in.

g) (Optional) In the `locality_name` field, replace `string` with the name of the city that your organization is located in.

h) (Optional) In the `organizational_unit_name` field, replace `string` with the name of your department within your organization.

The Value section should look similar to the following example:

```
{
  "subject": {
    "common_name": "example.com",
    "email_address": "admin@example.com",
    "organization_name": "Example",
    "country_code": "US"
  },
  "subject_alternative_names": [
    {
      "name": "www.example.com",
      "type": "dns"
    }
  ]
}
```

7. Click **Send Request** to create the signing request.
   In the Server response section, the Response body displays the signing request in the `pem` field.

Next steps

Send the signing request to your CA to create your signed SSL certificate.

Important: The signing request contains escape sequences that represent line breaks (\n). Replace each instance of \n with a line break before sending the request to your CA. You can modify the PEM request manually in a text editor or automatically through a JSON parsing utility, as shown in the following example command:

```
echo '<json_output>' | python -c 'import sys, json; print
 json.load(sys.stdin)["pem"]'
```

Replace the `<json_output>` variable with the entire JSON string returned in the Response Body section.

Add a trusted SSL certificate to your sensor or console

You can add an SSL certificate signed by a trusted CA to your sensor or console 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 **ExtraHop** and then click **PUT/extrahop/sslcert**.
5. Click **Try it out**.
6. In the **Certificate and Key** field, paste the SSL certificate.

   The certificate should look similar to the following text:

```
-----BEGIN CERTIFICATE-----
a0O8zvV4MlDhWX4e0VyvGAJx+9d4AqQB4Czy/P7z36CmHe2Y7PPdVSeWHNCQoJ0g
```

```
CnO42u2V9YKNFYRQejIJv8CxGVJKsdfV0iP0WnCvpZXkaBOYIrDvE5xn010WPUls
6qe3mCXsUK87i++mYuVDA1U0A5YVXRO2OOWIWy7P+MCU/cR/op3Jpekng2cxN4qD
FqGbtRpLdCuJ/xGWL1FFRHBg76+TbO+pxgZhiCtHYXfMKIaoPmDwsAqEtLbizz1W
mbMig9hs4QNcJ+aMNSnTZpkbeBR4a2nkGnQoYvnFOXV/nWzvfHmI4ydSH9g4I8qt
4ArqFepInvm70n07FYAKL6Mdd1i+7ieo9AqckltVzzKFzkakHm04214wtsYmle94
4HqIJ7p7NH5maXxttXMzHFlArbnjHWCl0gIv8lAu+IvLJ8aiGAb3zqveNz6ZAZ5j
PGAUsP+dVYV/8VjvqhkiP/1jWzUHwzpdlHbcD8qOkAF41fnbv+2EXqFJ096JSSiU
rqeJpgNuH3LbkT0KORAiLoGLMZKEKxF+3OpLVD7ox7NQh9pMdZlB8tcTbTmsvD8T
3L2tMVZssqYOANcidtd17t72VW4hzQURT1me5tGWxpN6od/q6B+FIvRq/7Vq0UE1
c2AG/om5UN/Vj3pUjXzq/B1IWUS9TicRcKdl5wrKEkPUGjK4w1R/87bj5HSn8nyd
lMCcOpLTokHj0B5+8O1ylNhVXNPlj3eY0n6OQOdClBqTDM0/4sB3XgeC/pjpleU3
3uot+wM/GoN/Dqb1LPt3BNpUQuCzSfmGSSOXiWELsEhz3ix/36a9eUWjfhmtPsW5
dne5Lf+G7cf+ebsRTb7R89GmgKzTpUl1KAzKINAebkT6WrWWljugpA0BcfANjS6o
mik4ZbY8d54UtA17evprr2+8UotIgVIrCbfLgA2DY8QOTCBYIFKJ3GZAedqRK9Sm
I2qdaB6QBczYNaVYSeCsBdHHw1+h7dBeqdUUwYKtmPW96/djj/6vJSXh9/UX/3c0
eqXG36w/lqJAYu8QtAydJsVC85IzqzikkX0f0KE315Doginpg59yix9dHD2sxLb1
X39BRpLkZ9nvW6ke2YHU/VKBVIxqSslukGoTUIcUtPJrtMQOwCi/EQQXbPK9a2pW
K51938h6OuLjNbDTFuxfhE4zITWHTgyAs2MNVR9+uDUiVJclX+CIPjhZzjyPqmD6
6uh8Sr3zndOMabqDquo69rMQyvclF0xOUMVgUw1Rb8Y=
-----END CERTIFICATE-----
```

Note: If you want the certificate to be signed with your own private key, you can include your key after the SSL certificate, separated by a line break. However, we recommend that you do not specify your own key; by default, the sensor or console will sign the certificate with the private key on the system.

7. Click **Send Request** to add the certificate.

Create custom devices through the REST API

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. 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 in to the sensor 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 through the REST API Explorer

You can create a custom device and associate the custom device with a list of IP addresses or CIDR blocks through the **POST /customdevices** operation.

Important: The REST API Explorer is not available on Reveal(x) 360.

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

   The URL is the hostname or IP address of your sensor, followed by `/api/v1/explore/`. For example, if your hostname is seattle-eda, the URL is `https://seattle-eda/api/v1/explore/`.
2. Click **Custom Device**, and then click **POST /customdevices**.
3. In the body field, specify properties for the custom device that you want to create.

   For example, the following body matches the custom device to the CIDR blocks 192.168.0.0/26, 192.168.0.64/27, 192.168.0.96/30, and 192.168.0.100/32:

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

```
  "name": "Seattle",
  "criteria": [
    {
      "ipaddr": "192.168.0.0/26"
    },
    {
      "ipaddr": "192.168.0.64/27"
    },
    {
      "ipaddr": "192.168.0.96/30"
    },
    {
      "ipaddr": "192.168.0.100/32"
    }
  ]
}
```

Retrieve and run the example Python script

The ExtraHop GitHub repository contains an example Python script that creates custom devices by reading criteria from a CSV file.

1. Go to the ExtraHop code-examples GitHub repository and download the `create_custom_devices/create_custom_devices.py` file to your local machine.
2. Create a CSV file with rows that contain the following columns in the specified order:

Name	ID	Description	IP address or CIDR block

Tip: The `create_custom_devices` directory contains an example CSV file named `device_list.csv`.

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.

3. In a text editor, open the `create_custom_devices.py` file and replace the following configuration variables with information from your environment:
   - **HOST:** The IP address or hostname of the sensor.
   - **APIKEY:** The API key.
   - **CSV_FILE:** The path of the CSV file relative to the location of the script file.
4. Run the following command:

```
python3 create_custom_devices.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)
```

Create and assign a device tag through the REST API

The following Python script creates a device tag and then assigns that tag to all of the devices in a specified subnet.

```
#!/usr/bin/env python

import httplib
import urllib
import json
import sys

# Configuration Options:
host = "{HOST}"
apikey = "{API KEY}"
tag_name = "MyTestTag"
subnet = "10.20.0.[0-9]+"
batch_limit = 100
headers = {'Accept': 'application/json',
           'Authorization': "ExtraHop apikey=%s" % apikey}
conn = httplib.HTTPSConnection(host)
  def execute_req(method, path, expected_code, failure_message, body=None):
"""
   Returns the body of a successful request,
   otherwise prints error and terminates
"""

conn.request(method, "/api/v1" + path, headers=headers, body=body)
resp = conn.getresponse()
if resp.status is not expected_code:
    print(failure_message)
    print(resp.read())
    sys.exit(1)
return resp

def execute_get(path, expected_code, failure_message):
    resp = execute_req("GET", path, expected_code, failure_message)
    return json.loads(resp.read())

def execute_create(path, body, expected_code, failure_message):
    """Returns ID of newly created resource"""
    resp = execute_req("POST", path, expected_code, failure_message, body)
    resp.read() # drain the response
    return int(resp.getheader("location").split("/")[-1])

# First, search for the specified tag, by name
resp = execute_get("/tags", 200, "Unable to retrieve tags from ExtraHop")
tags = [tag for tag in resp if tag["name"] == tag_name]

if not tags:
    # tag is not found, create it
    body = json.dumps({"name": tag_name})
    tag_id = execute_create('/tags', body, 201, "Unable to create tag")
else:
    tag_id = tags[0]["id"]

query_params = {'limit': batch_limit,
                'search_type': 'ip address',
                'value': subnet}
query_string = urllib.urlencode(query_params)

# Paginate device results, building up a list of all devices to assign
```

```
device_ids = []
offset = 0

while True:
    path = "/devices?" + query_string + ("&offset=%d" % offset)
    resp = execute_get(path, 200, "Unable to retrieve devices")
    if not resp:
        break

    device_ids += [device["id"] for device in resp]
    offset += batch_limit

# Perform the assignments
resp = execute_req("POST", "/tags/%d/devices" % tag_id,
                   204, "Unable to perform assignments",
                   body=json.dumps({"assign": device_ids}))
resp.read() # drain the response

# Check that assignments were successful
resp = execute_get("/tags/%d/devices" % tag_id,
                   200, "Unable to retrieve tag assignments")
assigned_device_ids = [device["id"] for device in resp]

successful = set(device_ids).issubset(set(assigned_device_ids))
if successful:
    print("%d devices assigned to tag" % len(device_ids))
else:
    print("Unable to assign all devices to tag")
```

Query for metrics about a specific device through the REST API

The following Python script queries for metrics from an HTTP client device with the ID 9363 and prints the response.

```
import httplib

headers = {'Content-Type': 'application/json',
       'Accept': 'application/json',
       'Authorization': 'ExtraHop apikey={API KEY}'
body = r"""{
  "cycle": "auto",
  "from": -1800000,
  "until": 0,
  "metric_category": "http_client",
  "metric_specs": [
    {
       "name": "req"
    }
   ],
   "object_ids": [
      9363
   ],
   "object_type": "device"
}"""
conn = httplib.HTTPSConnection('{HOST}')
conn.request('POST', '/api/v1/metrics', headers=headers, body=body)
resp = conn.getresponse()
print resp.status, resp.reason
print resp.read()
```

The following response shows entries for the device with ID 9363:

```
{
 "date": "Thu, 19 Nov 2015 23:20:07 GMT",
 "via": "1.1 localhost",
 "server": "Apache",
 "vary": "Accept-Encoding",
 "content-type": "application/json; charset=utf-8",
 "cache-control": "private, max-age=0",
 "connection": "Keep-Alive",
 "content-encoding": "gzip",
 "keep-alive": "timeout=45, max=44",
 "content-length": "277"
}

{
 "stats": [
   {
    "oid": 9363,
    "time": 1447973460000,
    "duration": 30000,
    "values": [
     2
    ]
   },
   {
    "oid": 9363,
    "time": 1447973490000,
    "duration": 30000,
    "values": [
     0
    ]
   },
   {
    "oid": 9363,
    "time": 1447973520000,
    "duration": 30000,
    "values": [
      1
     ]
    },
    {
     "oid": 9363,
     "time": 1447973550000,
     "duration": 30000,
     "values": [
       2
      ]
     }
```

Create, retrieve, and delete an object through the REST API

This example shows how you can create and successfully retrieve information about a device tag. Then, after the device tags are deleted, the example shows how an attempt to retrieve information subsequently fails.

The following example shows how to create a device tag called my_test_tag.

```
curl -i -X POST --header "Content-Type: application/json" \
--header "Accept: application/json" \
--header "Authorization: ExtraHop apikey={API KEY}" \
```

```
-d "{
\"name\": \"my_test_tag\"
}" "https://{HOST}/api/v1/tags"
```

A 201 status returns upon success with the following response headers, which display that the tag was created, and provides the device tag location and ID of /api/v1/tags/1.

```
{
 "date": "Wed, 18 Nov 2015 20:24:13 GMT",
 "via": "1.1 localhost",
 "server": "Apache",
 "content-type": "text/plain; charset=utf-8",
 "location": "/api/v1/tags/1",
 "cache-control": "private, max-age=0",
 "connection": "Keep-Alive",
 "keep-alive": "timeout=45, max=88",
 "content-length": "0"
}
```

Next, the ID (1) is added to the following GET request, which returns a 200 status upon success and the JSON representation of the retrieved tag:

```
curl -i -X GET --header "Accept: application/json" \
--header "Authorization: ExtraHop apikey={API KEY}" \
"https://{HOST}/api/v1/tags/1"
{
 "mod_time": 1447878253953,
 "id": 1,
 "name": "my_test_tag"
}
```

Next, the following example shows a DELETE request to remove the device tag from the system, which returns a 204 status upon success:

```
curl -i -X DELETE --header "Accept: application/json" \
--header "Authorization: ExtraHop apikey={API KEY}" \
"https://{HOST}/api/v1/tags/1"
```

Finally, when another GET request is sent for that deleted device tag, the operation fails, and a 404 status is returned upon failure, indicating that the tag is no longer available.

```
curl -i -X GET --header "Accept: application/json" \
--header "Authorization: ExtraHop apikey={API KEY}" \
"https://{HOST}/api/v1/tags/1"
```

Query the record log

The following request body queries the record log to retrieve 100 HTTP records where the method is GET and the status code is 404.

```
{
  "filter": {
    "operator": "and",
    "rules": [
        {
            "field": "method",
            "operand": "GET",
            "operator": "="
        },
```

```
        {
            "field": "statusCode",
            "operand": "404",
            "operator": "="
        }
    ]
 },
 "from": -900000,
 "limit": 100,
 "types": [
   "~http"
 ]
}
```