Extract metrics through the REST API

Published: 2019-02-15

You can extract metrics from an ExtraHop Discover or Command appliance through the REST API to visualize metrics in a third-party tool or compare ExtraHop data with other data you have collected. To extract a metric, you must first get identifiers for both the metrics you want to extract and the objects you want to extract metrics for. You can then build and test a metric query in the REST API Explorer before incorporating your request into a script that can read the metrics into a format that can be imported into applications.

Before you begin

- You must log into the ExtraHop appliance with an account that has unlimited privileges to generate an API key.
- You must have a valid API key to make changes through the REST API and complete the procedures below. (See Generate an API key.)
- Familiarize yourself with the ExtraHop REST API Guide to learn how to navigate the ExtraHop REST API Explorer.

Retrieve metric IDs

Metrics are identified in the ExtraHop REST API through a combination of the `metric_category`, the `name`, and the `object_type`. You can retrieve all three identifiers through the Metric Explorer.

1. Log into the Web UI on the Discover or Command appliance.
2. Click the System Settings icon and then click **Metric Catalog**.
3. In the Type to filter field, type the name of the metric you want to extract and then click the name of the metric in the search results below.
4. In the right pane, scroll down to REST API parameters and record the values.

   For example, the following information is displayed for the HTTP server responses metric:

REST API Parameters

```
{
    "metric_category": "http_server",
    "object_type": "device",
    "metric_specs": [
        {
            "name": "rsp"
        }
    ]
}
```

Retrieve object IDs

Next, you must find the unique identifier for the object that you want to extract metrics for in the REST API. You can retrieve this ID through the REST API Explorer.

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

The URL is the hostname or IP address of your ExtraHop Discover or Command appliance, followed by `/api/v1/explore/`. For example, if your hostname is seattle-eda, the URL is `https://seattle-eda/api/v1/explore/`.

2. Paste or type your API Key into the api_key field at the top of the page.
3. Click the type of object you want to collect metrics for, such as **Device**, **Device Group**, or **Application**.
4. Click **GET /<objects>**.

   For example, if you are extracting metrics for a device group, click **Get/ Device Groups**.
5. Optional: In the value field, type search criteria for the object you want to locate.

   For example, you can search for object names, IP addresses, or MAC addresses. If you are having difficulty locating a device in the ExtraHop system, see Find a device.
6. Click **Try it out!**.

   The Response Body displays information about each object that matches the search criteria.
7. Note the number in the id field for the object you want to collect metrics for.

   For example, the id of the following server is 1298:


```
[
  {
    "mod_time": 1516639693474,
    "node_id": null,
    "id": 1298,
    "extrahop_id": "fff4c3090a0a0000",
    "discovery_id": "fff4c3090a0a0000",
    "display_name": "server1",
    "description": null,
    "user_mod_time": 1512688149084,
    "discover_time": 1498685400000,
    "vlanid": 0,
    "parent_id": 140,
    "macaddr": "A1:01:01:01:1A:01",
    "vendor": "Mellanox",
    "is_l3": true,
    "ipaddr4": "10.10.10.200",
    "ipaddr6": null,
    "device_class": "node",
    "default_name": "Mellanox 10.10.10.200",
    "custom_name": "server1",
    "cdp_name": "",
    "dhcp_name": "server1.company.com",
    "netbios_name": "",
    "dns_name": "server1.company.com",
    "custom_type": "",
    "analysis_level": 1,
    "activity": []
  }
]
```


Query for metrics

You can query metrics through the REST API Explorer to make sure you have configured the right request body before adding the request to a script.

1. In the REST API Explorer, click **Metrics**, and then click **POST /metrics**.
2. In the body field, specify the metric that you want to extract.

For example, the following body extracts five minute metrics on HTTP responses for a server with an ID of 1298:

```
{
  "metric_category": "http_server",
  "metric_specs": [
    {
      "name": "rsp"
    }
  ],
  "object_type": "device",
  "object_ids": [
    1298
  ],
  "cycle": "5min"
}
```

The body must include the following parameters:

- **object_type:** The type of object you want to collect metrics for.
- **object_ids:** The id of the object you want to extract metrics for.
- **metric_category:** The category of the metric you want to collect.
- **name:** The name of the metric you want to collect.
- **cycle:** The aggregation period for metrics.

3. Click **Try it out!** to send the request to your appliance.
   The Response Body displays the requested metrics in JSON format.

Python Script Example

The following example Python script extracts the total count of HTTP responses a server with an ID of 1298 sent over five minute time intervals and then writes the values to a csv file:

```
import httplib
import json
import csv
import time

HOST = 'example.extrahop.com'
APIKEY = 'f6876657888a7c1f24ac77827'

headers = {'Content-Type': 'application/json',
           'Accept': 'application/json',
           'Authorization': 'ExtraHop apikey=%s' % APIKEY}
body = r"""{
  "metric_category": "http_server",
  "metric_specs": [
    {
      "name": "rsp"
    }
  ],
  "object_type": "device",
  "object_ids": [
   1298
  ],
  "cycle": "1hr"
}"""

conn = httplib.HTTPSConnection(HOST)
```

```
conn.request('POST', '/api/v1/metrics', headers=headers, body=body)
print "Extracting metrics"
resp = conn.getresponse()
parsed_resp = json.loads(resp.read())

output_file = 'output.csv'
with open(output_file, 'w') as csvfile:
    csvwriter = csv.writer(csvfile,dialect='excel')
    header = []
    v = 0
    for metric in parsed_resp['stats'][0]:
        header.append(metric)
    csvwriter.writerow(header)
    for metric in parsed_resp['stats']:
        metric['time'] = time.strftime('%Y-%m-%d %H:%M:%S',
 time.localtime(metric['time']/1000))
        metric['values'] = str(metric['values'][0])
        v += 1
        csvwriter.writerow(metric.values())
    print "Extracted %s metrics from %s to %s" %
 (str(v),parsed_resp['stats'][0]['time'],parsed_resp['stats'][-1]['time'])
```

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 appliance. Alternatively, you can add the context option to send the request over TLSv1.2 and bypass certificate verification; however, this method is not secure and is not recommended. The following code creates an HTTP connection over TLSv1.2:

```
conn = httplib.HTTPSConnection(HOST,
 context=ssl.SSLContext(ssl.PROTOCOL_TLSv1_2))
```