Query for records through the REST API

Published: 2020-02-23

The ExtraHop REST API enables you to query for records stored on an ExtraHop explore appliance. By querying records with a REST API script, you can import records into a third party application, such as Microsoft Excel. Also, if your query matches more than the maximum number of records returned by the REST API, you can configure the script to recursively query for the remaining records. In this topic, we show methods for querying records through both the ExtraHop REST API Explorer and a Python script.

Before you begin

- You must log into the ExtraHop appliance with an account that has full write 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.

Query records 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. 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 **Record Log** and then click **POST /records/search**.
5. Click **Try it out**.
   The JSON schema is automatically added to the body parameter text box.
6. In the body text box, specify fields for your record query.
   For example, the following fields retrieve records from the last 30 minutes that include an IP address, domain name, or URI that has been identified as suspicious according to threat intelligence collections on an appliance:

```
{
    "from": "-30m",
    "filter": {
        "field": "ex.isSuspicious",
        "operator": "=",
        "operand": {
            "type": "boolean",
            "value": "true"
        }
    }
}
```

   For a complete list of valid fields, see the Body Parameters section under **POST /records/search** in the REST API Explorer.

Python script example

The following python script queries for records over the last 30 minutes that involve an IP address, domain name, or URI that has been identified as suspicious according to threat intelligence collections on an appliance. The script then and writes specified record fields to a CSV file that can be read by Microsoft Excel.

Note: For more information about threat intelligence with ExtraHop, see Threat intelligence - Reveal(x) only and Upload STIX files through the REST API to Reveal(x).

The script includes the following configuration variables that you must replace with information from your environment:

- **HOST:** The IP address or hostname of the Discover appliance. Note that this hostname is not the hostname of the connected Explore appliance that the records are stored on.
- **APIKEY:** The API key.
- **FILENAME:** The file that output is written to.
- **TIME_LIMIT:** If the record query matches more than 100 records, the amount of time after the initial query that the remaining records can be retrieved from the appliance.
- **QUERY:** The record query parameters.
- **COLUMNS:** The record fields that are written to the CSV output file.

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

import json
import requests
import unicodecsv as csv

HOST = 'example.extrahop.com'
API_KEY = 'f6876657888a7c1f24ac77827'
FILENAME = "records.csv"
TIME_LIMIT = '1m'
QUERY = {
    "context_ttl": TIME_LIMIT,
    "from": "-30m",
    "filter": {
        "field": "ex.isSuspicious",
        "operator": "=",
        "operand": {
            "type": "boolean",
            "value": "true"
        }
    }
}
COLUMNS =
 ['timestamp','sender','senderAddr','senderPort','receiver','receiverAddr','receiverPort

# Method that performs an initial record query on an ExtraHop appliance
def recordQuery(query):
    url = HOST + '/api/v1/records/search'
    headers = {'Authorization': 'ExtraHop apikey=%s' % API_KEY}
    r = requests.post(url, headers=headers, data=json.dumps(query))
    try:
        return json.loads(r.text)
    except:
        print 'Record query failed'
        print r.text
        print r.status_code

# Method that retrieves remaining records from a record query
```

```
def continueQuery(cursor):
    url = HOST + '/api/v1/records/cursor'
    headers = {'Authorization': 'ExtraHop apikey=%s' % API_KEY}
    query = {'cursor': cursor}
    r = requests.post(url, headers=headers, data=json.dumps(query))
    try:
        return json.loads(r.text)
    except:
        print 'Record query failed'
        print r.text
        print r.status_code

# Query records from appliance
response = recordQuery(QUERY)
records = response['records']
if 'cursor' in response:
    response_cursor = response['cursor']
    retrieved = len(records)
    while retrieved > 0:
        print 'Retrieved  ' + str(len(records)) + ' of ' +
 str(response['total']) + ' total records'
        response = continueQuery(response_cursor)
        newRecords = response['records']
        retrieved = len(newRecords)
        records = records + newRecords
print 'Total records retrieved = ' + str(len(records))

# Simplify and format records for CSV
table = []
for record in records:
    row = {}
    fields = record['_source']
    for column in COLUMNS:
        try:
            value = fields[column]
            # Retrieve isSuspicious field from ex object
            if column == 'ex':
                try:
                    row['isSuspicious'] = value['isSuspicious']
                except:
                    row[column] = value
            # Concatenate values returned as lists
            elif type(value) is list:
                row[column] = ' '.join(value)
            # Retrieve values from dict objects
            elif type(value) is dict:
                try:
                    # If value is a list, concatenate list
                    if type(value['value']) is list:
                        row[column] = ' '.join(value['value'])
                    else:
                        row[column] = value['value']
                except:
                    row[column] = value
            else:
                row[column] = value
        except:
            row[column] = ''
    table.append(row)


# Write records to csv
with open(FILENAME, 'w') as csvfile:
    csvwriter = csv.writer(csvfile,dialect='excel', encoding='utf-8')
```

```
    csvwriter.writerow(table[0].keys())
    for row in table:
        csvwriter.writerow(row.values())
```

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 `verify=False` option to bypass certificate verification; however, this method is not secure and is not recommended. The following code sends an HTTP GET request without certificate verification:

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