Query for records through the REST API

Published: 2024-02-23

The ExtraHop REST API enables you to query for records stored on a recordstore. 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 in to the sensor or console 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 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 **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:

```
{
    "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 examples

The following Python scripts query for records that involve an IP address, domain name, or URI that has been identified as suspicious according to threat intelligence. The scripts then write specified record fields to a CSV file that can be viewed in a spreadsheet program.

Note: For more information about threat intelligence with ExtraHop, see Threat intelligence and Upload STIX files through the REST API.

Retrieve and run the example Python script for an ExtraHop recordstore

The ExtraHop GitHub repository contains an example Python script that retrieves records from an ExtraHop recordstore.

Important: If the query matches more than the maximum number of records that can be retrieved at once, the script retrieves the remaining records by sending a cursor to the sensor or console with the POST /records/cursor operation. This operation is only valid with ExtraHop recordstore. If you have configured a third-party or cloud recordstore, see Retrieve and run the example Python script for a third-party or cloud recordstore.

1. Go to the ExtraHop code-examples GitHub repository and download the `query_records_explore/query_records_explore.py` file to your local machine.
2. In a text editor, open the `query_records_explore.py` file and replace the following configuration variables with information from your environment:

   • **HOST:** The IP address or hostname of the sensor or console. Note that this hostname is not the hostname of the connected ExtraHop recordstore 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 system.

   • **QUERY:** The record query parameters.

   • **COLUMNS:** The record fields that are written to the CSV output file.
3. Run the following command:

```
python3 query_records_explore.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)
```

Retrieve and run the example Python script for a third-party or cloud recordstore

The ExtraHop GitHub repository contains an example Python script that retrieves records from third-party and cloud recordstores.

Note: If the query matches more than the maximum number of records that can be retrieved at once, the script retrieves the remaining records by sending additional requests with the `offset` parameter. The offset parameter skips a specified number of records in a query.

1. Go to the ExtraHop code-examples GitHub repository and download the `query_records_third_party/query_records_third_party.py` file to your local machine.
2. In a text editor, open the `query_records_third_party.py` file and replace the following configuration variables with information from your environment:

   • **HOST:** The IP address or hostname of the sensor or console.

   • **APIKEY:** The API key.

   • **FILENAME:** The file that output is written to.

- **LIMIT:** The maximum number of records to retrieve at a time.
- **QUERY:** The record query parameters.
- **COLUMNS:** The record fields that are written to the CSV output file.

3. Run the following command:

```
python3 query_records_third_party.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)
```