

Upgrade ExtraHop firmware through the REST API

Published: 2024-03-26

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.	<ul style="list-style-type: none"> • Command appliance • Discover appliances • All Explore appliances (manager nodes, then data nodes)
Explore appliances	See Upgrading ExtraHop recordstores .	<ul style="list-style-type: none"> • Trace appliances
Trace appliances	None	

Upgrade ExtraHop firmware through the REST API Explorer

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 `ebdbbc9e-7113-448c-ab9b-cc0ec2307702`

```
/api/v1/jobs/ebdbbc9e-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 `ebdbbc9e-7113-448c-ab9b-cc0ec2307702`:

```
< Location: /api/v1/jobs/ebdbbc9e-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": true}'
```