Cloudera Edge Management 1.6.0

Using Asset Push Command

Date published: 2019-04-15
Date modified: 2023-07-27

CLOUDERA

https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Using Asset Push command.....4

Updating agent capability using Asset Push from EFM.....6

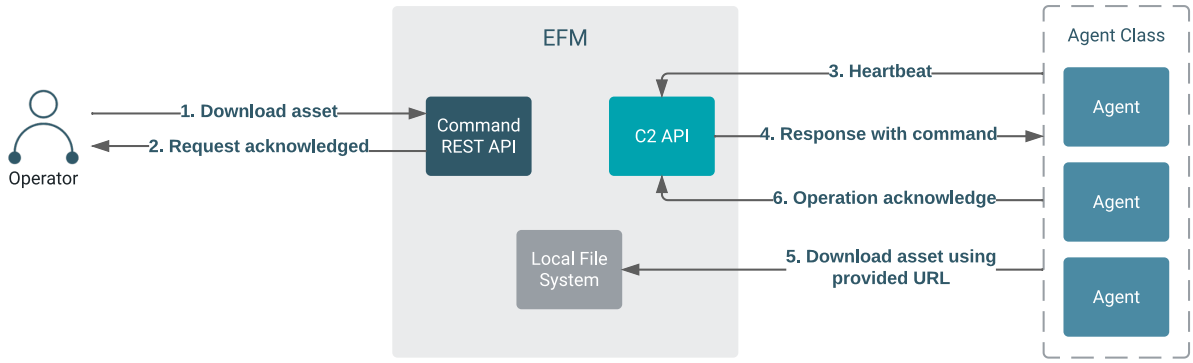
Using Asset Push command

Learn how you can trigger an asset push command to agents of a particular agent class from the Swagger UI.

You need to initiate the process by issuing an `UPDATE ASSET` command against an agent class through the REST API. The call is asynchronous and returns the created operation details. On the next heartbeat, the agent receives the operation (command) in the heartbeat response. The agent initiates the download through the provided URL which points to EFM. EFM proxies the URL to the respective external storage. External storage is a local filesystem or S3. Asset files have to be uploaded to S3 or deployed on all EFM nodes under the same path and name. This path is passed on as a parameter for the request so EFM will use it to identify the asset. As there is one incoming request and there could be multiple EFM nodes, they should have the exact same setup. After the asset is downloaded, the agent acknowledges the operation.

Asset push for asset in local file system

In case of an asset located in the local file system, the workflow of the process is the following:



Running the asset push command

1. Ensure that the asset is available on all EFM nodes, which the agents download.
2. Construct the request payload.

   For example:

   ```
   {
       "assetFileName": "test.txt",
       "assetUri": "/tmp/input.txt",
       "forceDownload": false
   }
   ```

   where

   - assetFileName = The target file name to be used on the agent.
   - assetUri = Location of the asset on EFM.
   - forceDownload = Forces the agent to skip all version checks and perform asset download.
3. Send a POST request to /efm/api/commands/<testAgentClass>/update-asset with the payload created in the previous step and the agent class name.

   This triggers all agents under the agent class to download the requested asset.

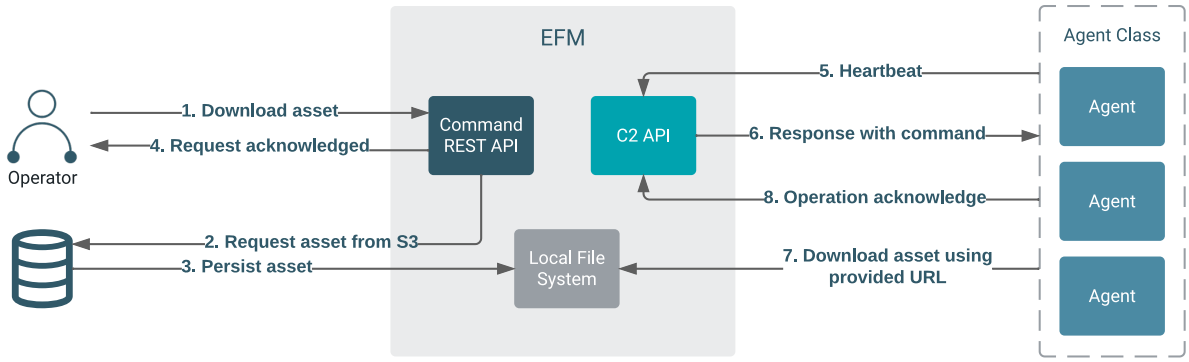
4. Go to the Edge Events page and check for the `UPDATE ASSET` command.

   You can see the `UPDATE ASSET` command where you can track how it goes through its lifecycle stages (for example, queued, deployed, and done). You can also check the command's current state under the Commands tab in the Agent Details page for the Agent Manager.

   Result: After the commands are executed, assets are available on all agents belonging to the specified agent class.

Asset push for asset located in S3

In case of an asset located in S3, the workflow of the process is the following:



Running the asset push command

1. Ensure that the asset is available on S3.
2. Construct the request payload.

   Example 1:

```
{
 "assetFileName": "test-asset.txt",
 "assetUri": "s3://test-asset-bucket-1/test-asset.txt",
 "forceDownload": true,
 "customProperties": {
  "accessKeyId": "***",
  "secretAccessKey": "***",
  "region": "us-west-2"
 }
}
```

   Example 2:

```
{
 "assetFileName": "test-asset.txt",
 "assetUri": "s3://test-asset-bucket-1/test-asset.txt",
 "forceDownload": true,
 "customProperties": {
  "accessKeyId": "***",
  "secretAccessKey": "***",
  “sessionToken”: “***”,
  "region": "us-west-2"
 }
}
```

   where

   - assetFileName = The target file name to be used on the agent.
   - assetUri = Location of the asset on S3.

- forceDownload = Forces the agent to skip all version checks and perform asset download.
- customProperties = Defines the AWS credentials that EFM will use during asset download from S3.

3. Send a POST request to /efm/api/commands/<testAgentClass>/update-asset with the payload created in the previous step and the agent class name.
   - First this triggers all EFM nodes to download the asset from S3 to the local file system.
   - After that EFM will triggers all agents under the agent class to download the requested asset.
4. Go to the Edge Events page and check for the `UPDATE ASSET` command.

   You can see the `UPDATE ASSET` command where you can track how it goes through its lifecycle stages (for example: queued, deployed, and done). You can also check the command's current state under the Commands tab on the Agent Details page of the Agent Manager.

   Result: When the process is completed, assets are available on all agents belonging to the specified agent class.

Agent configuration

On the agent side the default location where the asset will be saved is ${MINIFI_HOME}/asset. You can customize this path with the c2.asset.directory property.

Restrictions and limitations

These are current limitations of asset push:

- If asset push fails, there is no automated recovery or retry option.
- Assets must be located on all EFM nodes or in S3. It also means that the hardware capabilities of your EFM node(s) can be a limitation for download speed or storage capacity for assets.
- Required agent versions:
  - CEM MiNiFi C++ Agent - 1.22.06+
  - CEM MiNiFi Java Agent - 1.23.02+

Updating agent capability using Asset Push from EFM

Learn how you can use Asset Push to automate the process of distributing NAR files to the agents much faster.

About this task

MiNiFi Java is released with a set of core processors and controller services. As the flow or requirements evolve, it can happen that a more niche or a custom processor is required. In this case all MiNiFi agents would have to be updated independently to add the new capability. Combining property update with asset push and manifest refresh, everything can be done directly from EFM without accessing each agent separately.

You can watch this demo video about the process:

Or just follow the below steps to update agent capabilities remotely:

Procedure

1. Gather all NAR files that are required for your processor.

   Some processors require multiple NARs. You can either create your custom processor and bundle it as NARs or you can use the Cloudera Artifactory for available NARs: https://repository.cloudera.com/artifactory/repo/org/apache/nifi/

   Note: Cloudera Artifactory contains Java NARs, which only work with MiNiFi Java Agents.

2. Upload the NAR file(s) to S3 or all EFM nodes.

   For more information, see *Using Asset Push command*.

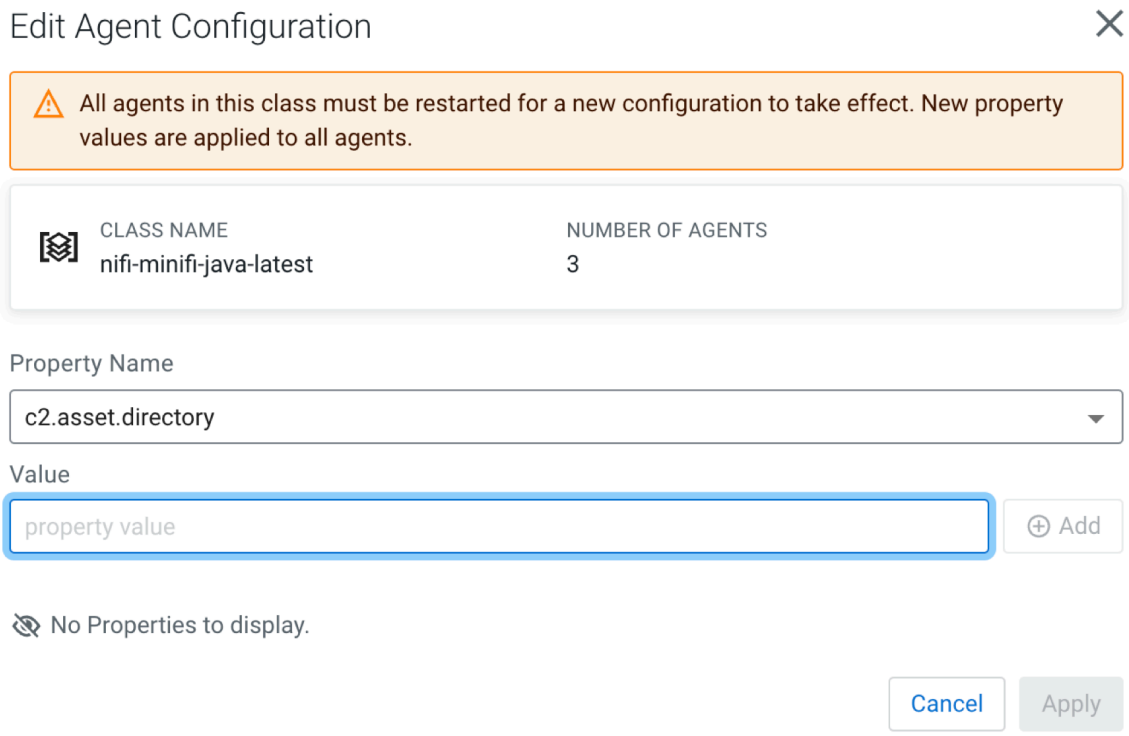
   Note: EFM tries to serve these requests from in-memory cache. If you exceed the default 16 MB asset size, your asset push operation could increase I/O operations on EFM nodes. You can fine tune these settings with the efm.asset.cache.maxSizeByte property.

3. Set the value of the asset directory property to the extension directory, which is *./extensions* by default.
   - Java agent property: c2.asset.directory
   - C++ agent property: nifi.c2.asset.directory

   On the UI:

   a. Click Monitor on the left navigation panel to open the Dashboard.
   b. Select an agent class.
   c. Click Actions Edit Agent Configuration .

   The Edit Agent Configuration dialog appears:

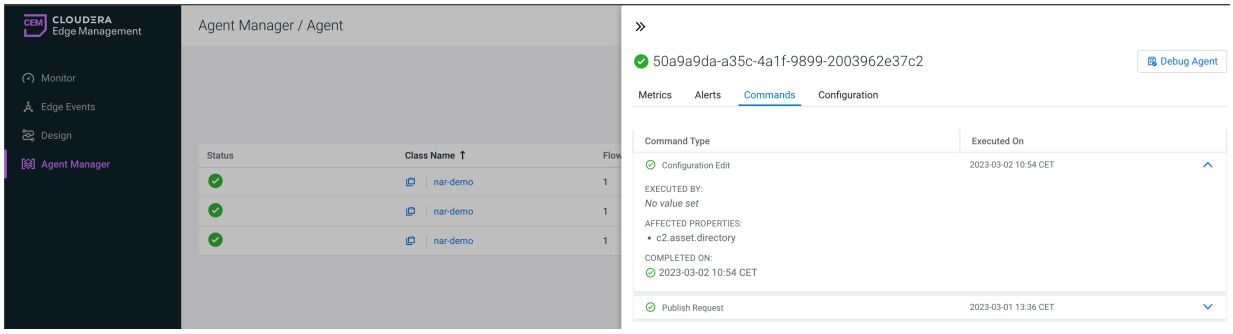


   d. Select the asset directory property (for a Java agent, it is c2.asset.directory).
   e. Add *./extensions* in the Value field, and click Apply.

   Through Swagger:

   http:/<cem-host>:<cem-port>/efm/swagger/#/Commands/createPropertyUpdateOperation

   You can check the command status in the Agent Manager:

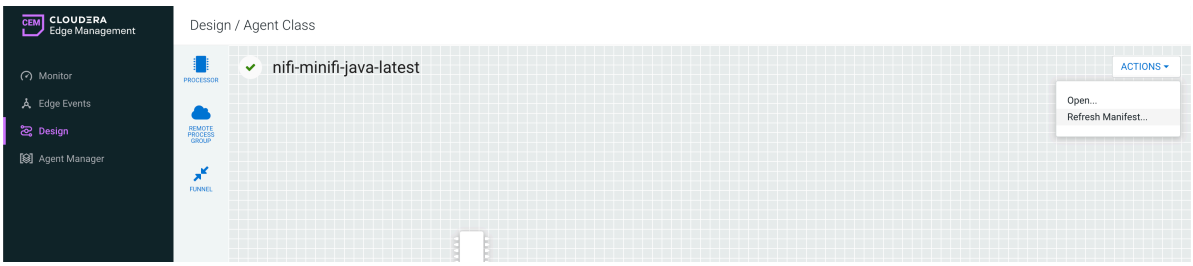


4. Run an asset push.
   - For instructions, see *Using Asset Push command*.
   - Through Swagger: use

     http://<cem-host>:<cem-port>/efm/swagger/#/Commands/createUpdateAssetOperation
   - You can check the command status in the Agent Manager. You have to wait an extra heartbeat interval to complete manifest update.

   Important: If you have multiple NARs, you have to execute the asset push for each NAR.

5. Refresh the agent manifest.
   - For instructions, see *Refreshing the agent manifest for existing flows*.
   - On the UI:

   - Through Swagger: use

     http://<cem-host>:<cem-port>/efm/swagger/#/Flow%20Designer/refreshFlowManifest

Results

The new processors/controller services are now available. You can check them on the EFM UI.

Related Information

Using Asset Push command

Refreshing the agent manifest for existing flows