HCP Upgrade Guide 1

Hortonworks Cybersecurity Platform

Date of Publish: 2018-07-30

http://docs.hortonworks.com

Contents

Preparing to Upgrade

Hortonworks Cybersecurity Platform (HCP) upgrades are not officially supported. However you can use the guidelines provided in the Upgrade Guide if you want to attempt an upgrade. Prior to upgrading Hortonworks Cybersecurity Platform (HCP), you must back up your configuration and stop all Metron services.

Back up Your Configuration

The Hortonworks Cybersecurity Platform (HCP) upgrade uses the default configuration for the new Metron version. If you made any changes to the Metron configuration in the previous version, you must back up your old configuration so you can incorporate those changes into the new Metron configuration. You will also need to re-enter values for the Metron properties in Ambari.

Procedure

1. Create a backup directory.

```
mkdir /$HCP_BACKUP_DIRECTORY
```

2. Back up your configuration information in ZooKeeper to your backup directory:

```
${METRON_HOME}/bin/zk_load_configs.sh -m DUMP -z $ZOOKEEPER >
/$HCP_BACKUP_DIRECTORY/$BACKUP_CONFIG.txt
```

3. Back up the following property files in the $METRON_HOME/config directory to your backup directory:

   - elasticsearch.properties
   - enrichment.properties
   - pcap.properties

   For example:

```
cp elasticsearch.properties /$HCP_BACKUP/elasticsearch.properties
```

4. Copy the zookeeper directory to your backup directory:

```
cp -R zookeeper/ /$HCP_BACKUP/zookeeper
```

5. Back up your Metron configuration.

   The easiest way to do this is to take a screenshot of each of the Metron configuration pages that you modifed in Ambari. At a minimum, take a screen shot of the following configuration pages:

   - Index Settings
   - Parsers
   - REST

Stop All Metron Services

You need to stop all Metron services prior to uninstalling Metron.

Procedure

1. Stop all Metron services in Ambari.

   Stop each Metron service in the following order:

- Metron Alerts UI
- Metron Management UI
- Metron REST

2. Stop Storm:
   a) From the Storm node, list all of the Storm topologies that are currently running:

   ```
   storm list
   ```

   b) Kill each of the running Storm topologies in the following order:

   - all parsers such as bro and snort
   - enrichment
   - indexing
   - profiler

   ```
   storm kill bro
   ```

   c) Return to the Storm UI and verifiy that all topologies are killed.
   d) In Ambari, stop Storm by selecting Storm and clicking **Stop All** in the **Actions** menu.
3. Ensure that the UIs are shut down.

   If the Metron Alerts Ui or Metron Management UI status in Ambari is "running," shut down the UIs by entering the following from $METRON_HOME/var/log/metron/metron:

   ```
   service metron-alerts-ui status
   service metron-alerts-ui stop

   service metron-management-ui status
   service metron-management-ui stop
   ```

Upgrade Metron

After you shut down Metron and all of its services, you must uninstall Metron and then reinstall the newest version of Metron.

Before you begin

- Back up your Metron configuration.
- Stop all Metron services

Procedure

1. Uninstall Metron.

   In Ambari, select **Metron**, then under the **Service Actions** menu, click **Delete Service**.

   When prompted, enter "delete" to confirm deleting the service.
2. Remove all of the rpms from the old Metron version.

   CentOS

   a) From the Ambari node, enter the following to list all of the Metron packages:

   ```
   rpm -qa | grep metron
   ```

You should see input similar to the following:

```
metron_1_4_2_0_23-config-0.4.1.1.4.2.0-23.noarch
```

b) Enter the following to list all of the Metron packages:

```
sudo rpm -q --scripts  metron_1_4_2_0_23-config-0.4.1.1.4.2.0-23.noarch
```

You should see output similar to the following:

```
chkconfig --add metron-management-ui
chkconfig --add metron-alerts-ui
preuninstall scriptlet (using /bin/sh):
chkconfig --del metron-management-ui
chkconfig --del metron-alerts-ui
```

c) Remove each of the package:

```
rmp remove $PACKAGE_NAME
```

For example:

```
sudo chkconfig --del metron-management-ui
```

Ubuntu

From the Ambari node, enter the following to delete all of the Metron packages:

```
sudo aptitude purge $PACKAGE_NAME
```

3. Modify the /etc/yum.repos.d/HCP.repo file with the updated repo version:

```
vi /etc/yum.repos.d/HCP.repo
```

4. Update the HCP.repo file.

CentOS

```
yum update
```

Ubuntu

```
apt-get update
```

5. Install the current HCP mpack repo from Release Notes.

```
wget http://public-repo-1.hortonworks.com/HCP/centos7/1.x/updates/1.4.1.0/
tars/metron/hcp-ambari-mpack-1.4.1.0-18.tar.gz
ambari-server install-mpack --force --mpack=/${MPACK_DOWNLOAD_DIRECTORY}/
hcp-ambari-mpack-1.4.1.0-18.tar.gz --verbose
```

6. Restart the Ambari server.

```
ambari-server restart
```

7. Re-open Ambari and add back the updated Metron version.

From the **Actions** menu, click **Add Service**, then click Metron from the **Choose Services** page. Ensure Metron is the updated version.

Ambari lists each service on which Metron is dependent.

8. Click yes to add each dependency.

9. In Ambari, add back your Metron configuration information in the **Property** fields.

   Do not copy and paste into the Metron property fields. You can inadvertently add a special character.

10. Click **Deploy** to start the Metron set up.

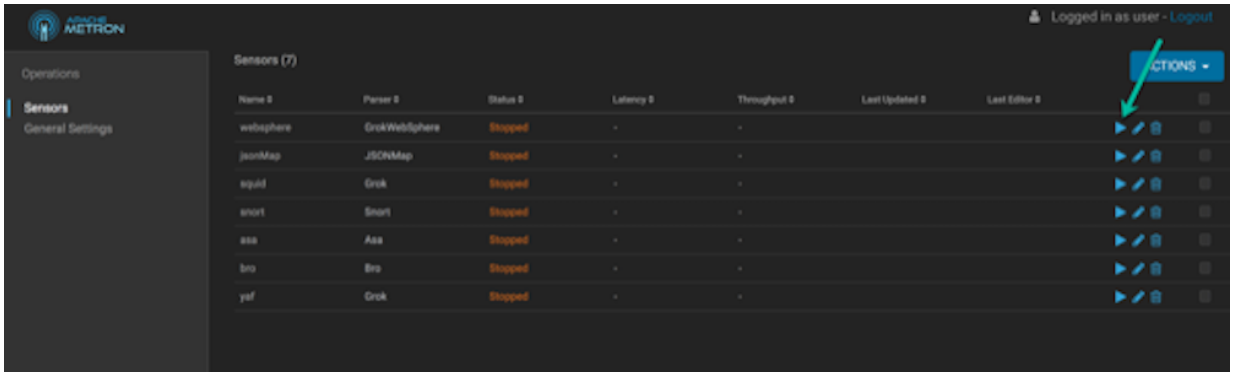
    The process to install, start, and test Metron will take a while.

11. Restart the Metron services:

    - Metron REST
    - Metron Management UI
    - Metron Alerts UI
    - Indexing

12. In the Management UI, restart the Metron Parsers including Enrichment, Bro, Snort, Yaf, and any other parsers you added previously.

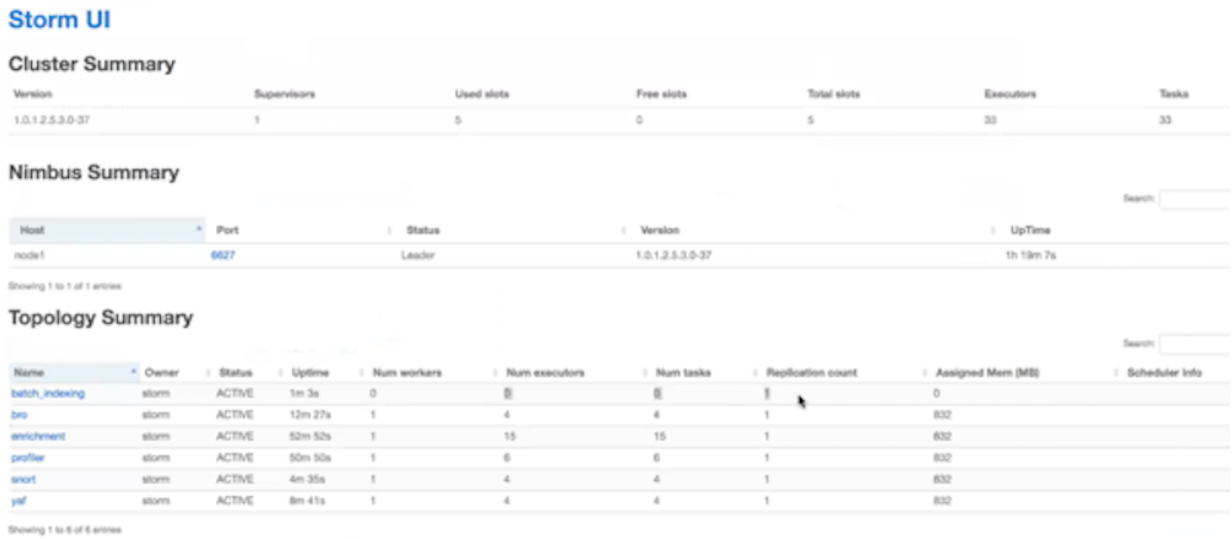
    Management UI



    Note: Starting the Metron parsers might take a while.

13. Check the status of the parsers in the Storm UI.

    Storm UI



Mandatory Post-Upgrade Tasks

After you finish updating the Ambari M-Pack, depending on your configuration, you need to update the various features in your cluster.

Upgrading Your Configuration

Hortonworks Cybersecurity Platform (HCP) upgrade uses the default configuration for the new Metron version. If you made any changes to the Metron configuration in the previous version, you must incorporate those changes into the new Metron configuration.

Changes to the Metron configuration can effect the following:

- Metron properties in Ambari
- ZooKeeper

  Incorporate changes from the ZooKeeper file you backed up earlier.
- Flux files

  Incorporate changes from the Flux files you backed up earlier.

Changes to STELLAR Language

Hortonworks Cybersecurity Platform (HCP) adds additional Stellar keywords to each new HCP version. These new keywords might cause compatablity issues where these reserved words and symbols are used in existing scripts.

Check the Stellar Language Quick Reference for new and changed Stellar keywords.

HCP 1.4.3 adds match to the Stellar lanaguage which introduces the following new reserved keywords and symbols:

match, default, {, }, '=>'

You must modify any Stellar expressions that use these keywords not in quotes.

Upgrading to Elasticsearch 5.6.2

There are a number of template changes in Elasticsearch 5.6.2, most notably around string type handling, that may cause issues when upgrading.

For Elasticsearch 5.x, the existing indexes and templates need to upgraded. For more information, see:

- Updating Elasticsearch Templates
- Updating Existing Indexes

If you are upgrading from Elasticsearch 2.x to Elasticsearch 5.6.2, you will need to re-index.

Type Mapping Changes

Type mappings in Elasticsearch 5.6.2 have changed from ES 2.x. This section provides an overview of the most significant changes.

The following is a list of the major changes in Elasticsearch 5.6.2:

- String fields replaced by text/keyword type
- Strings have new default mappings as follows:

```
{
  "type": "text",
  "fields": {
    "keyword": {
      "type": "keyword",
      "ignore_above": 256
    }
  }
```

```
}
```

- There is no longer a _timestamp field that you can set "enabled" on.

  This field now causes an exception on templates. The Metron model has a timestamp field that is sufficient.

The semantics for string types have changed. In 2.x, index settings are either "analyzed" or "not_analyzed" which means "full text" and "keyword", respectively. Analyzed text means the indexer will split the text using a text analyzer, thus allowing you to search on substrings within the original text. "New York" is split and indexed as two buckets, "New" and "York", so you can search or query for aggregate counts for those terms independently and match against the individual terms "New" or "York." "Keyword" means that the original text will not be split/analyzed during indexing and instead treated as a whole unit. For example, "New" or "York" will not match in searches against the document containing "New York", but searching on "New York" as the full city name will match. In Elasticsearch 5.6 language, instead of using the "index" setting, you now set the "type" to either "text" for full text, or "keyword" for keywords.

Below is a table listing the changes to how String types are now handled.

sort, aggregate, or access values	Elasticsearch 2.x	Elasticsearch 5.x	Example
no	`"my_property" : { "type": "string", "index": "analyzed" }`	`"my_property" : { "type": "text" }` Additional defaults: "index": "true", "fielddata": "false"	"New York" handled via in-mem search as "New" and "York" buckets. No aggregation or sort.
yes	`"my_property": { "type": "string", "index": "analyzed" }`	`"my_property": { "type": "text", "fielddata": "true" }`	"New York" handled via in-mem search as "New" and "York" buckets. Can aggregate and sort.
yes	`"my_property": { "type": "string", "index": "not_analyzed" }`	`"my_property" : { "type": "keyword" }`	"New York" searchable as single value. Can aggregate and sort. A search for "New" or "York" will not match against the whole value.
yes	`"my_property": { "type": "string", "index": "analyzed" }`	`"my_property": { "type": "text", "fields": { "keyword": { "type": "keyword", "ignore_above": 256 } } }`	"New York" searchable as single value or as text document, can aggregate and sort on the sub term "keyword."

If you want to set default string behavior for all strings for a given index and type, you can do so with a mapping similar to the following (replace ${your_type_here} accordingly):

```
# curl -XPUT 'http://${ES_HOST}:${ES_PORT}/_template/
default_string_template' -d '
{
    "template": "*",
    "mappings" : {
        "${your_type_here}": {
            "dynamic_templates": [
                {
                    "strings": {
                        "match_mapping_type": "string",
                        "mapping": {
                            "type": "text"
                            "fielddata": "true"
                        }
                    }
                }
            ]
        }
    }
}
```

By specifying the template property with value *, the template will apply to all indexes that have documents indexed of the specified type (${your_type_here}).

The following are other settings for types in ES:

- doc_values
  - On-disk data structure
  - Provides access for sorting, aggregation, and field values
  - Stores same values as _source, but in column-oriented fashion better for sorting and aggregating
  - Not supported on text fields
  - Enabled by default
- fielddata
  - In-memory data structure
  - Provides access for sorting, aggregation, and field values
  - Primarily for text fields
  - Disabled by default because the heap space required can be large

Troubleshooting

If you run into issues with your upgrade use the following troubleshooting tips to identify and resolve those issues.

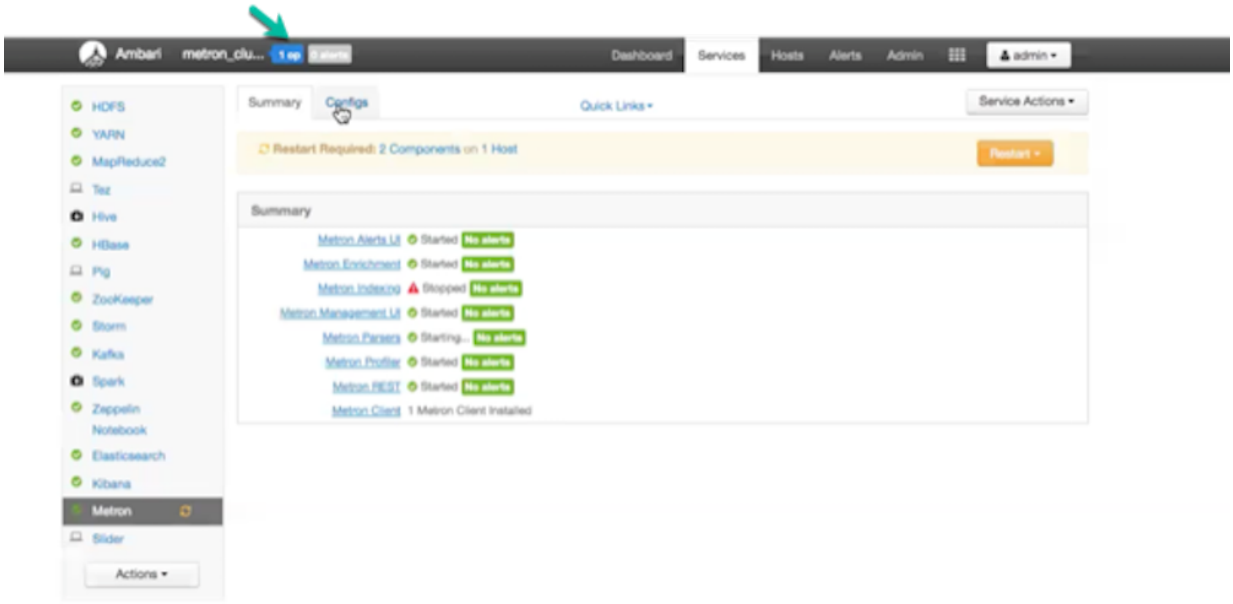
Checking the Status of the Parsers

If your parsers do not restart, you can use Ambari to check the status of the parsers and restart them.

Procedure

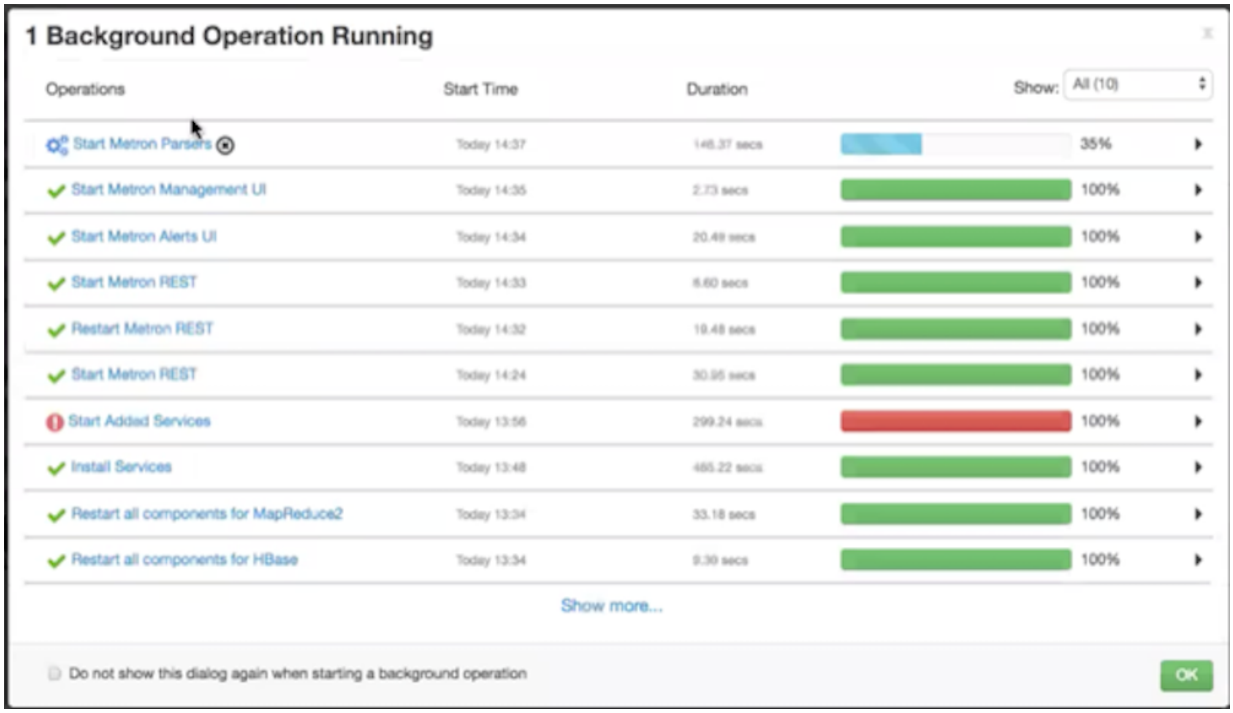
1. Click the operation status tab at the top of the Ambari window.

   Ambari Summary Tab



Ambari displays the Operations Running Status window.

Ambari Background Operation Page



2. Click **Start Metron Parsers**.
   Ambari displays the **Start Metron Parsers** window.
3. Click the parser node you want to check, then click **Metron Parsers Start**.
   Ambari displays information on the status of the parser.

   Metron Parsers Start Page

4. Review the information in this window to determine the status of your parsers.