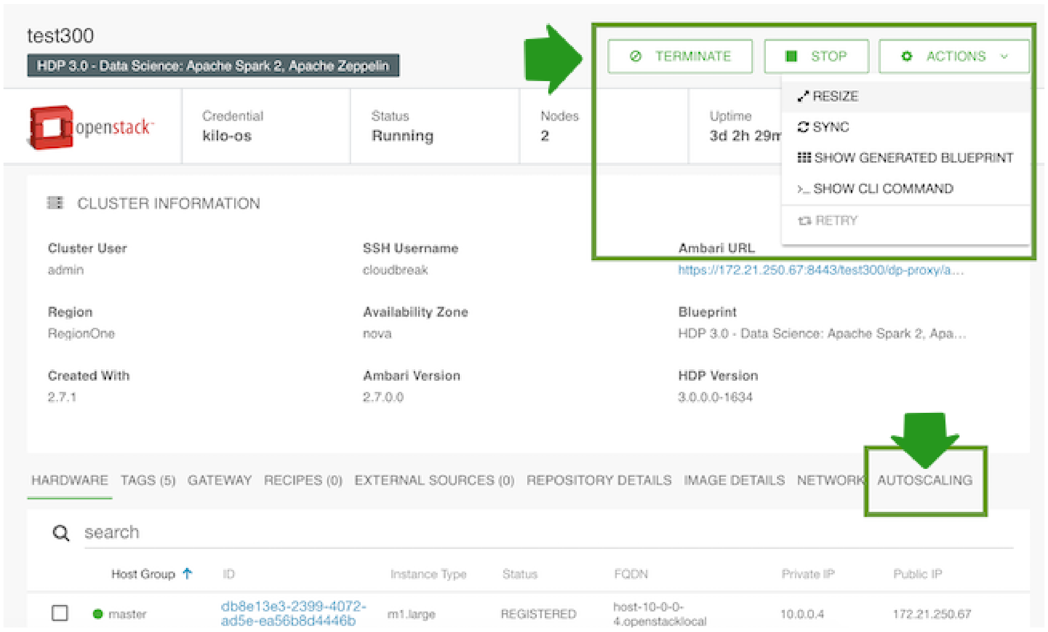
Managing Clusters

Date of Publish: 2019-05-28

Contents

Managing and monitoring clusters

You can manage monitor your clusters from the Cloudbreak web UI. To do that, click on the tile representing the cluster that you want to access. The actions available for your cluster are listed in the top right corner:



Credential: kilo-os
Status: Running
Nodes: 2
Uptime: 3d 2h 29m

Cluster User admin	SSH Username cloudbreak	Ambari URL https://172.21.250.67:8443/test300/dp-proxy/a...
Region RegionOne	Availability Zone nova	Blueprint HDP 3.0 - Data Science: Apache Spark 2, Apa...
Created With 2.7.1	Ambari Version 2.7.0.0	HDP Version 3.0.0.0-1634

	Host Group	ID	Instance Type	Status	FQDN	Private IP	Public IP
☐	● master	db8e13e3-2399-4072-ad5e-ea56b8d4446b	m1.large	REGISTERED	host-10-0-0-4.openstacklocal	10.0.0.4	172.21.250.67

Retry a cluster

When stack provisioning or cluster creation fails, the “retry” option allows you to resume the process from the last failed step.

In some cases the cause of a failed stack provisioning or cluster creation can be eliminated by simply retrying the process. For example, in case of a temporary network outage, a retry should be successful. In other cases, a manual modification is required before a retry can succeed. For example, if you are using a custom image but some configuration is missing, causing the process to fail, you must log in to the machine and fix the issue; Only after that you can retry the process.

Only failed stack or cluster creations can be retried. A retry can be initiated any number of times on a failed creation process.

To retry provisioning a failed stack or cluster, follow these steps.

Steps

1. Browse to the cluster details.
2. Click Actions and select Retry.

   Only failed stack or cluster creations can be retried, so the option is only available in these two cases.
3. Click Yes to confirm.

   The operation continues from the last failed step.

Resize a cluster

The resize option allows you to add or remove cluster nodes.

To resize a cluster, follow these steps.

Note:

Cluster resizing is not supported for HDF clusters.

Steps

1. Browse to the cluster details.
2. Click Actions and select Resize. The cluster resize dialog is displayed.
3. Using the +/- controls, adjust the number of nodes for a chosen host group.

   Note:

   You can only modify one host group at a time. It is not possible to resize the Ambari server host group.
4. Click Yes to confirm the scale-up/scale-down.

   While nodes are being added or removed, cluster status changes to “Update In Progress”. Once the operation has completed, cluster status changes back to “Running”.

Synchronize a cluster

The sync option allows you to synchronize Cloudbreak with the cloud provider and with Ambari.

Use the sync option if you:

- Made changes on your cloud provider side (for example, deleted cluster VMs) and you would like to synchronize Cloudbreak with the cloud provider.
- Manually changed service status in Ambari (for example, restarted services).

To synchronize your cluster with the cloud provider, follow these steps.

Steps

1. Browse to the cluster details.
2. Click Actions and select Sync.
3. Click Yes to confirm.

   Your cluster infrastructure is synchronized based on changes on the cloud provider. The updates are written to the “Event History”.

Stop a cluster

Cloudbreak supports stopping and restarting clusters. Once a cluster is in the ""Running" state, it can be stopped.

Steps

1. Browse to the cluster details.
2. Click Stop to stop a currently running cluster.
3. Click Yes to confirm.
4. Your cluster status changes to “Stopping in progress” and then to “Stopped”. Once stopping the infrastructure has completed, you will see a Start option to restart your cluster.

When a cluster is in the “stopped” state, cluster VMs are given back to the cloud provider; Once the cluster is restarted, new VMs are provisioned. Therefore when a cluster is in the “stopped” state, you are no longer charged for VMs, but you are charged for external storage.

Restart a cluster

If your cluster is in the “Stopped” state, you can restart it by using the "Start" option.

Steps

1. Click Start. This option is only available when the cluster has been stopped.
2. Click Yes to confirm.

   Your cluster status changes to “Start in progress” and then to “Running”.

Terminate a cluster

You can terminate any cluster managed by Cloudbreak from that Cloudbreak instance.

Steps

1. Browse to the cluster details.
2. Click Terminate.
3. Click Yes to confirm.

   All cluster-related resources will be deleted, unless the resources (such as networks and subnets) existed prior to cluster creation or are used by other VMs in which case they will be preserved.

Force terminate a cluster

The force terminate option can be used when cluster deletion fails.

Cluster deletion may fail if Cloudbreak is unable to delete one or more of the cloud resources that were part of your cluster infrastructure. Use the Terminate > Force terminate option to remove the cluster entry from the Cloudbreak web UI. If you use this option, you must also check your cloud provider account to see if there are any resources that must be deleted manually.

Steps

1. Browse to the cluster details.
2. Click Terminate.
3. Check Force terminate.
4. Click Yes to confirm.

   When terminating a cluster with Kerberos enabled, you have an option to disable Kerberos prior to cluster termination. This option removes any cluster-related principals from the KDC.
5. This deletes the cluster tile from the UI.
6. Log in to your cloud provider account and manually delete any resources that failed to be deleted.

Related Information

Deleting clusters

Deleting clusters when termination fails

Upon cluster termination, Cloudbreak only terminates the resources that it created. It does not terminate any resources (such as networks, subnets, roles, and so on) which existed prior to cluster creation.

The proper way to delete clusters is by using the Terminate option available from the cluster details in the Cloudbreak web UI. If the terminate process fails, try the Terminate > Force terminate option. If the force termination does not delete all cluster resources, delete the resources manually:

- To find the VMs, click on the links available in the Hardware section in the cluster details.

  If you terminate instances on the cloud provider without Cloudbreak, during the sync operation, Cloudbreak realizes that instances have been removed and marks the instances as DELETED_ON_PROVIDER_SIDE.

On AWS, GCP, and OpenStack, Cloudbreak automatically removes all VM dependencies such as attached disks, network interfaces, public IPs and storage accounts. On Azure, you must manually delete the disks created for that instance and release the public IP assigned to that instance.
- To find the network and subnets, see the Cluster Information pane in the cluster details.

Repairing a cluster

Cloudbreak monitors clusters, ensuring that when host-level failures occur, they are quickly resolved by deleting and replacing failed nodes along with their attached volumes.

For each cluster, Cloudbreak checks for Ambari agent heartbeat on all cluster nodes. If the Ambari agent heartbeat is lost on a node, a failure is reported for that node. This may happen for the following reasons:

- The ambari-agent process exited on a node
- An instance crashed
- An instance was terminated

Once a failure is reported, it is repaired automatically (if auto repair is enabled), or options are available for you to repair the failure manually (if auto repair is disabled).

When a cluster is fault-tolerant (HA cluster), all nodes can be repaired, including the Ambari server node. In case of a cluster without fault tolerance (non-HA cluster), all nodes can be repaired except the Ambari server node.

Warning:

In order to be able to use the repair feature, you need to set up custom hostnames. This is required because Ambari stores hostname-related metadata in its database and it requires custom hostnames to keep consistent metadata. You can set this up by using either Custom internal hostnames for cluster hosts or Custom hostnames based on DNS on AWS.

Auto repair

If auto repair is enabled, once a failure of a worker or compute node is reported, it is repaired automatically be removing and replacing the failed node. The flow is:

1. A notification about node failure is printed in the UI.
2. The recovery flow is triggered. Cluster status changes to 'REPAIR'.
3. Downscale: Remove failed nodes, copy data from volumes attached to the failed nodes to other volumes, and then remove the attached volumes.
4. Upscale: New nodes and attached volumes of the same type are added in place of the failed nodes).
5. The recovery flow is completed. The cluster status changes to 'RUNNING'.

Corresponding events are written to cluster’s event history.

Manual repair

Manual repair is enabled for all clusters by default. When manual repair is enabled, when a worker or compute node fails:

1. A notification about node failure is printed in the UI:
   - Cluster tile on the cluster dashboard shows unhealthy nodes
   - Nodes are marked as "UNHEALTHY" in the Hardware section
   - Cluster's event history shows "Manual recovery is needed for the following failed nodes"
2. You have an option to repair or delete failed nodes. This option is available from the cluster details in the UI (Actions > Repair) and from the CLI (the cluster repair command).
3. If repair was chosen: (1) Failed nodes are removed and data from volumes attached to the failed nodes in copied to other volumes, and then the attached volumes are removed. (2) New nodes and attached volumes of the same type are added in place of the failed nodes.

4. If delete was chosen, failed nodes are deleted with their attached volumes.
5. Once the recovery flow is completed, the cluster status changes to 'RUNNING'.

Corresponding events are written to cluster’s event history.

Enabling auto or manual repair

For each cluster, you can enable auto or manual repair. Manual repair is default.

Warning:

In order to be able to use the repair feature, you need to set up custom hostnames. This is required because Ambari stores hostname-related metadata in its database and it requires custom hostnames to keep consistent metadata. You can set this up by using either Custom internal hostnames for cluster hosts or Custom hostnames based on DNS on AWS.

Manual repair is used by default for all clusters. If you would like to use auto repair, set “recoveryMode”:”MANUAL” in the CLI JSON. That is, the following repair options are available:

- Manual: “recoveryMode”:”MANUAL” (default)
- Auto: “recoveryMode”:”AUTO”

Additionally, when an HA blueprint is provided, setting auto repair can be done from the web UI. The option is available from the Hardware and Storage page > Enable Auto Recovery. If “Enable Auto Recovery” is not selected, manual recovery is configured by default. This option is not available when using a non-HA blueprint.

Performing manual repair

Manual repair should be performed on a cluster that has nodes marked as unhealthy.

When a cluster has unhealthy nodes, a warning is displayed:

- Cluster tile on the cluster dashboard shows unhealthy nodes
- Nodes are marked as "UNHEALTHY" in the Hardware section
- Cluster's event history shows "Manual recovery is needed for the following failed nodes"

If manual repair has been enabled for the cluster, perform manual repair from the web UI or CLI.

To perform manual repair from the web UI:

1. To repair a cluster, select Actions > Repair from cluster details.
2. Select the host group that should be repaired. Only one host group can be selected at a time.
3. By default, unhealthy nodes are removed and then replaced. If you would like to just remove the nodes without replacing them, select Remove only.
4. Click Repair.

To perform manual repair from the CLI, use the following commands:

- cb cluster list – Check the status and health of your clusters
- cb cluster describe – Check the status and health of a specific cluster
- cb cluster repair – Perform cluster repair.

For more information, refer to CLI Reference documentation.

Note:

Recovery can fail during downscale on worker nodes if there is not enough space for HDFS to move data away from the volume attached to the failing node.

When repair fails on NAMENODE in an HA cluster

When repair fails on NAMENODE in an HA cluster, follow these steps to fix the cluster.

In such cases, there is a set of manual steps which have to be executed to resolve the problem and bootstrap the namenode:

1. Access the newly launched repaired host via SSH.
2. Run the following commands:

```
su hdfs
hdfs namenode -bootstrapStandby
```

3. Synchronize your cluster.

Configure autoscaling

Autoscaling allows you to adjust cluster capacity based on Ambari metrics and alerts, as well as schedule time-based capacity adjustment. Refer to this section if you would like to configure autoscaling for Cloudbreak-managed clusters.

When creating an autoscaling policy, you define:

- An alert that triggers a scaling policy. An alert can be based on an Ambari metric or can be time-based.
- A scaling policy that adds or removes a set number of nodes to a selected host group when the conditions defined in the attached alert are met.

Metric-based autoscaling

Cloudbreak accesses all available Ambari metrics and allows you to define alerts based on these metrics. For example:

Alert Definition	**Policy Definition**
ResourceManager CPU alert with CRITICAL status for 5 minutes	Add 10 worker nodes
HDFS Capacity Utilization alert with WARN status for 20 minutes	Set the number of worker nodes to 50
Ambari Server Alerts alert with CRITICAL status for 15 minutes	Decrease the number of worker nodes by 80%

Time-based autoscaling

Time-based alerts can be defined by providing a cron expression. For example:

Alert Definition	**Policy Definition**
Every day at 07:00 AM (GMT-8)	Add 90 worker nodes
Every day at 08:00 PM (GMT-8)	Remove 90 worker nodes

Note:

Cluster resizing is not supported for HDF clusters.

Overview of configuring autoscaling

1. Enable autoscaling.
2. Define a time-based or metric-based alert.
3. Create a scaling policy.
4. Review and, if needed, configure autoscaling settings.

Enable autoscaling

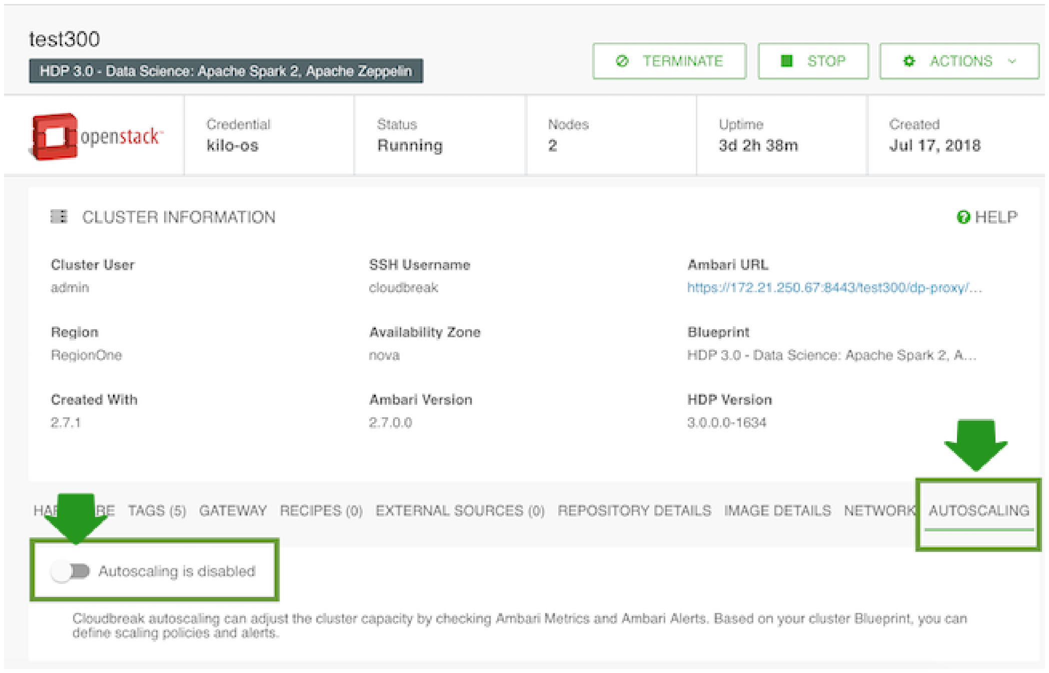
For each newly created cluster, autoscaling is disabled by default. You can enable it once a cluster is in the running state.

Note:

Autoscaling configuration is only available in the web UI. It is currently not available in the CLI.

Steps

1. On the cluster details page, navigate to the Autoscaling tab.
2. Click the toggle button to enable autoscaling:



test300

HDP 3.0 - Data Science: Apache Spark 2, Apache Zeppelin

Credential kilo-os
Status Running
Nodes 2
Uptime 3d 2h 38m
Created Jul 17, 2018

CLUSTER INFORMATION

Cluster User admin
SSH Username cloudbreak
Ambari URL https://172.21.250.67:8443/test300/dp-proxy/...
Region RegionOne
Availability Zone nova
Blueprint HDP 3.0 - Data Science: Apache Spark 2, A...
Created With 2.7.1
Ambari Version 2.7.0.0
HDP Version 3.0.0.0-1634

Autoscaling is disabled

Cloudbreak autoscaling can adjust the cluster capacity by checking Ambari Metrics and Ambari Alerts. Based on your cluster Blueprint, you can define scaling policies and alerts.

3. The toggle button turns green and you can see that "Autoscaling is enabled".
4. Define alerts and then define scaling policies. You can also adjust autoscaling settings.

If you decide to disable autoscaling, your previously defined alerts and policies will be preserved.

Define an alert

After enabling autoscaling, define a metric-based or time-based alert.

Define a metric-based alert

Perform the following steps to create a time-based alert.

Steps

1. In the Alert Configuration section, select the Time Based alert type.
2. Provide the following information:

Parameter	Description
Enter alert name.	Enter a unique name for the alert.
Select timezone.	Select your timezone.
Enter cron expression	Enter a cron expression that defines the frequency of the alert. Refer to Cron expression generator.

3. Click + to save the alert.

Once you have defined an alert, create a scaling policy that this metric should trigger.

Define a time-based alert

Perform the following steps to create a metric-based alert.

Steps

1. In the Alert Configuration section, select Metric Based alert type.
2. Provide the following information:

Parameter	Description
Enter alert name	Enter a unique name for the alert.
Choose metric type	Select the Ambari metric that should trigger the alert.
Alert status	Select the alert status that should trigger an alert for the selected metric. One of: OK, CRITICAL, WARNING.
Alert duration	Select the alert duration that should trigger an alert.

3. Click + to save the alert.

Once you have defined an alert, create a scaling policy that this metric should trigger.

Note:

If you would like to change default thresholds for an Ambari metric, refer to Ambari documentation.

If you would like to create a custom Ambari alert, refer to How to create a custom Ambari alert and use it for Cloudbreak autoscaling policies.

Related Information

Cron expression generator

Modifying alerts (Ambari)

How to create a custom Ambari alert and use it for Cloudbreak autoscaling policies (HCC)

Create a scaling policy

After enabling autoscaling and creating at least one alert, perform the following steps to create a scaling policy.

Steps

1. In the Policy Configuration section, provide the following information:

Parameter	Description
Enter policy name	Enter a unique name for the policy.
Select action	Select one of the following actions: Add (to add nodes to a host group) Remove (to delete nodes from a host group), or Set (to set the number of nodes in a host group to the chosen number).
Enter number or percentage	Enter a number defining how many or what percentage of nodes to add or remove. If the action selected is “set”, this defines the number of nodes that a host group will be set to after scaling.
Select nodes of percent	Select “nodes” or “percent”, depending on whether you want to scale to a specific number, or percent of current number of nodes.
Select host group	Select the host group to which to apply the scaling.
Choose an alert	Select the alert based on which the scaling should be applied.

2. Click + to save the alert.

Configure autoscaling settings

After enabling autoscaling, perform these steps to configure the auto scaling settings for your cluster.

Steps

1. In the Cluster Scaling Configuration, provide the following information:

Setting	Description	Default Value
Cooldown time	After an auto scaling event occurs, the amount of time to wait before enforcing another scaling policy. This means that the scaling events scheduled during cooldown time are dropped.	30 minutes
Minimum Cluster Size	The minimum size allowed for the cluster. Auto scaling policies cannot scale the cluster below or above this size.	2 nodes
Maximum Cluster Size	The maximum size allowed for the cluster. Auto scaling policies cannot scale the cluster below or above this size.	100 nodes

2. Click Save to save the changes.

Add SSL certificate for Knox Gateway

When Knox-powered Gateway is enabled, use these steps if you would like to add an SSL certificate for Ambari and/ or other cluster UIs exposed through the Gateway.

Steps

1. Obtain a trusted SSL certificate.
2. If needed, perform the following to make sure that your certificate is compatible with the Gateway:
   - If the certificate is not already in p12 format, export the certificate into p12 format. For example:

     ```
     openssl pkcs12 -export -in cacert.pem -inkey cakey.pem -out identity.p12
      -name gateway-identity -password pass:$mastersecret
     ```
   - Ensure that the certificate alias is “gateway-identity”.
   - Ensure that the store password matches the master secret created earlier. You can obtain the master secret of the Knox by using the following command:

     ```
     cat /srv/pillar/gateway/init.sls | grep mastersecret
     ```
   - Note the key password used – as you need to create an alias for this password.
3. Access the cluster's master node via ssh.
4. Obtain root access by using sudo su.
5. Use keytool to import the desired certificate/key pair into the java keystore that Knox is using.

   You can find the java keystore the following path:

   ```
   /usr/hdp/current/knox-server/data/security/keystores/gateway.jks
   ```

   Example command for importing your certificate into the jks:

   ```
   keytool -importkeystore -deststorepass $mastersecret -destkeypass
    $mastersecret -destkeystore gateway.jks -srckeystore /usr/hdp/
   current/knox-server/data/security/keystores/custom_certs/identity.p12 -
   srcstoretype PKCS12 -srcstorepass $mastersecret -alias gateway-identity
   ```
6. Restart Knox by using the following command:

   ```
   /usr/hdp/current/knox-server/bin/gateway.sh stop
   ```

This command stops Knox, but systemd automatically restarts it. To validate that it is restarted, use:

```
netstat -tlpn | grep 8443
```

Here is example output showing that the restart was successful:

```
netstat -tlpn | grep 8443
tcp  0  0  0.0.0.0:8443 0.0.0.0:*  LISTEN  13177/java
```

7. Using your web browser, access the Ambari web UI.
8. Confirm that the connection is SSL-protected and that the certificate used is the certificate that you provided.

Add SSL certificate for Ambari

By default Cloudbreak configures Ambari with a self-signed certificate for access via HTTPS. This is sufficient for many deployments such as trials, development, testing, or staging. However, for production deployments, you should obtain and configure a trusted certificate.

Determine which instructions to use

Depending on your configuration, use the following instructions for adding a trusted certificate for Ambari:

Scenario	Instructions
The Knox-powered Gateway is enabled and Ambari is exposed through the Gateway (default behavior).	Use Add SSL certificate for Knox Gateway
The Knox-powered Gateway is disabled or Ambari is not exposed through the Gateway.	Use the instructions provided in this section

Add SSL certificate for Ambari

Use these steps if you would like to add an SSL certificate for Ambari when Knox-powered Gateway is disabled.

Note:

Only use these instructions only if you disabled the Knox-powered Gateway for Ambari. When Knox Gateway is enabled and Ambari is exposed through it (default behavior), use the instructions provided in Add SSL certificate for Knox Gateway.

Steps

1. Obtain a trusted SSL certificate.
2. Access the Ambari server host via ssh.
3. Obtain root access by using sudo su.
4. Copy the certificate to the /etc/certs-user-facing/ directory on the master host.
5. Open the /etc/nginx/sites-enabled/ssl-user-facing.conf file for editing.
6. At server which listens on 443, update the ssl_certificate path and the ssl_certificate_key path to point to the location if the new certificate and key:

```
server {
    listen       443;
    ssl on;
    ssl_certificate      /etc/certs-user-facing/server.pem;
    ssl_certificate_key  /etc/certs-user-facing/server-key.pem;
...
```

7. Restart nginx. Depending on the nginx distribution, use of the following commands:

   ```
   systemctl restart nginx
   ```

   or

   ```
   service nginx restart
   ```
8. Using your web browser, access the Ambari web UI.
9. Confirm that the connection is SSL-protected and that the certificate used is the certificate that you provided.

Updating OS and tools on long-running clusters

Long-running clusters require maintenance updates such as updates to the base image's operating system and any third party packages that have been installed. Cloudbreak supports performing these maintenance updates.

In general, updating a cluster involves (1) manually updating OS and tools on existing nodes and (2) switching to a new image that will be used for any nodes added in the future via cluster scaling. The general steps are:

1. Preparing a new custom image (if using custom images) that includes the desired version of OS and tools.
2. Manually updating all existing cluster nodes to use the same OS and tools versions as the new image.
3. Setting that new image as default for all newly added cluster nodes.

These steps are described in more detail below.

The following limitations apply:

- You can only upgrade to a pre-warmed image if the Ambari/HDP/HDF versions are the same on the new image and on the old image.
- For base image only, the Saltboot and Salt versions must match on the new image and on the old image.
- In all cases, Linux OS must be the same. For example, you cannot upgrade from Redhat to Centos. Furthermore, the major X version must also be the same (i.e. you cannot update from version 6 to 7, but you can update from 6.1 to 6.2).

Step 1: Prepare a new custom image

If using custom images, prepare a new custom image with the desired package versions, and either update/extend your existing image catalog to reference the new image or prepare and then register a new image catalog. For more information about using custom images, refer to custom images documentation.

Step 2: Manually update existing cluster nodes

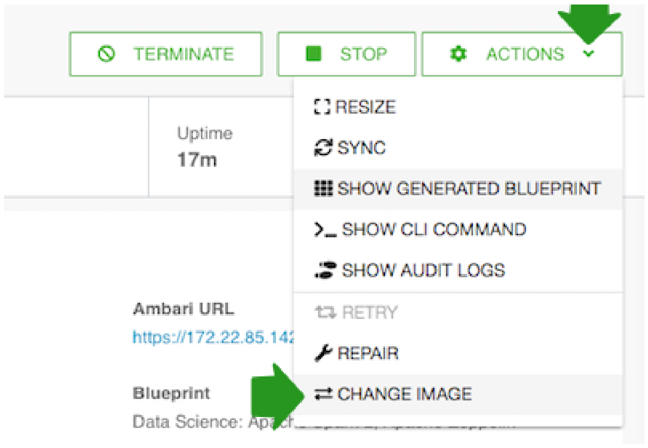
You should manually update all existing cluster nodes to use the same OS and tools versions as the new image. Perform these updates as described in your OS and tools documentation. Only after performing this step, you can proceed to the next step, which is updating cluster’s default image for nodes that will be added in the future.

Step 3: Update cluster’s default image

You should update your cluster to use the new default image for newly added cluster nodes. This can be done by using the web UI or CLI.

(Option 1: web UI) To update cluster’s default image by using the web UI:

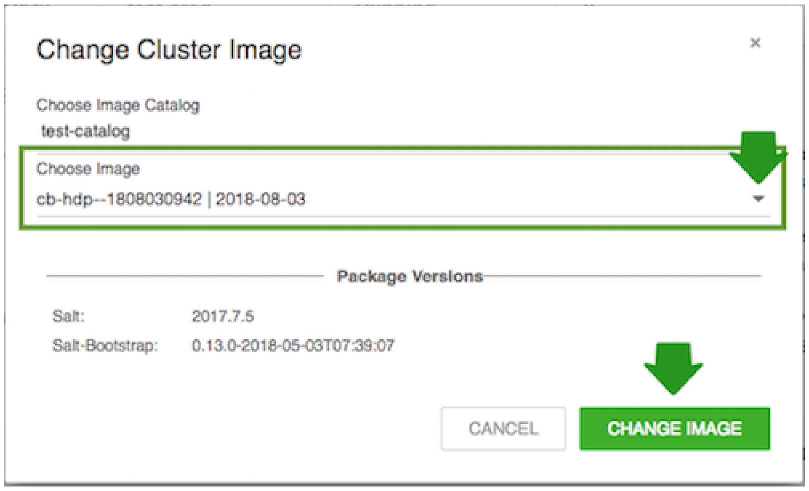
1. Navigate to the cluster details.
2. Sync the cluster (Actions > Sync) to ensure that Cloudbreak has the latest image and package information.
3. From the Actions menu select Change image.



4. Under Choose Image Catalog, choose the image catalog where the image that you would like to use is included.

   Note:

   If this window does not show any images, try



Change Cluster Image

Choose Image Catalog
test-catalog

Choose Image
cb-hdp--1808030942 | 2018-08-03

Package Versions

Salt:	2017.7.5
Salt-Bootstrap:	0.13.0-2018-05-03T07:39:07

5. Under Choose Image, select the image that you would like to use as the new default image. Click on Choose image.

This will set the new image as default for all newly added nodes.

Note:

This will not update currently running cluster nodes. Only the nodes added after the default image update will use the new default image. Existing cluster nodes, must be updated separately.

(Option 2: CLI) To update cluster's default image by using the CLI:

1. Sync the cluster:

```
cb cluster sync --name <cluster-name>
```

2. List the available images you can upgrade your cluster to:

```
cb imagecatalog images cluster-upgrade --cluster <cluster-name>
```

3. Update your cluster's default image:

```
cb cluster change-image --name <cluster-name> --imageid <image-id>
```

Related Information

Using custom images

Updating Ambari, HDP, and HDF on long-running clusters

Long-running clusters require maintenance updates such as Ambari and stack updates. Cloudbreak supports performing these maintenance updates.

In general, updating a cluster involves (1) upgrading Ambari and/or HDP/HDF on existing nodes and, if needed, (2) switching to a new default image that will be used for any nodes added in the future via cluster scaling. Ambari and/or HDP/HDF upgrade can be performed when your cluster has been put in maintenance mode.

The general steps are:

1. Enabling maintenance mode for the cluster that you are planning to update.
2. Upgrading Ambari and/or HDP/HDF.
3. Updating Ambari and/or HDP/HDF repo information.
4. Changing cluster's default image, if needed.
5. Validating the repository configuration.
6. Disabling maintenance mode.

Step 1: Enable maintenance mode

Prior to updating your cluster, you must enable maintenance mode for that cluster. This can be done from the Cloudbreak CLI by using the following steps:

1. On your computer, export the name of the cluster that you are attempting to upgrade. For example (replace test-cluster in the example with the actual cluster name):

   ```
   export clustername=test-cluster
   ```

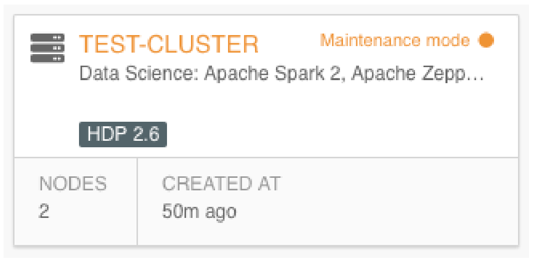
   Note:

   This is optional, but if you don’t perform this step, you should replace ${clustername} in all commands listed here with your actual cluster name.
2. Enable maintenance mode for the cluster that you would like to update. You can do this by using the following CLI command:

   ```
   cb cluster maintenance-mode enable --name "${clustername}"
   ```
3. To verify that maintenance mode is enabled, do one of the following:

   - Run the cb cluster list command and confirm that the cluster status is "MAINTENANCE_MODE_ENABLED":

     ```
     cb cluster list
     [
       {
          "Name": "test-cluster",
          "Description": "",
          "CloudPlatform": "OPENSTACK",
          "StackStatus": "AVAILABLE",
          "ClusterStatus": "MAINTENANCE_MODE_ENABLED"
        }
     ]
     ```
   - Verify in the web UI that the cluster tile looks similar to the following:



Once your cluster is in maintenance mode, only Sync, Terminate, and Change image actions are available. Other actions cannot be performed: Cloudbreak will return an error if you try to use them. Once your cluster is in maintenance mode, you can proceed to upgrading Ambari and HDP/HDF.

Step 2: Upgrade Ambari and HDP/HDF

Once your cluster is in maintenance mode, you can upgrade Ambari and/or stack. If updating both Ambari and stack, update Ambari prior to updating HDP/HDF.

1. Upgrade Ambari. If your cluster is in HA mode, make sure to update the Ambari server on each of the gateway nodes. For instructions, refer to Ambari upgrade documentation for the Ambari version that you would like to upgrade to.
2. Upgrade HDP or HDF. For instructions, refer to HDP or HDF upgrade documentation for the HDP or HDF version that would like to upgrade to.

Step 3: Update Ambari repo information

After the upgrade is complete, update Ambari repo information in Cloudbreak by using the following steps. If you only upgraded the stack without upgrading Ambari, skip this step.

The commands below use jq. To install jq, refer to https://stedolan.github.io/jq/download/.

1. (Optional) Check the current repository information by using cb cluster describe:

```
cb cluster describe --name "${clustername}" | jq
 ".cluster.ambariRepoDetailsJson"
```

2. Use the cb cluster maintenance-mode generate-template-json command to generate a JSON template:

```
cb cluster maintenance-mode generate-template-json ambari
```

3. Create a new JSON file, paste the generated template, and provide correct repository information by replacing the ____ with actual values of the updated Ambari or HDP/HDF repository.
4. Configure repository metadata by passing the JSON file created in the previous steps:

```
cb cluster maintenance-mode ambari --name "${clustername}" --cli-input-
json ~/Users/myuser/ambari.json
```

   Replace ~/Users/myuser/ambari.json with the actual name and location of your JSON file.
5. (Optional) Verify that the update was successful by using cb cluster describe:

```
cb cluster describe --name "${clustername}" | jq
 ".cluster.ambariRepoDetailsJson"
```

Step 4: Update HDP/HDF repo information

After the upgrade is complete, update stack repo information in Cloudbreak by using the following steps. If you only upgraded Ambari without upgrading the stack, skip this step.

The commands below use jq. To install jq, refer to https://stedolan.github.io/jq/download/.

1. (Optional) Check the current repository information by using cb cluster describe:

   ```
   cb cluster describe --name "${clustername}" | jq
    ".cluster.ambariStackDetails"
   ```
2. Use the cb cluster maintenance-mode generate-template-json command to generate a JSON template:

   For HDP:

   ```
   cb cluster maintenance-mode generate-template-json hdp
   ```

   For HDF:

   ```
   cb cluster maintenance-mode generate-template-json hdf
   ```
3. Create a new JSON file, paste the generated template, and provide correct repository information by replacing the ____ with actual values of the updated Ambari or HDP/HDF repository.
4. Configure repository metadata by passing the JSON file created in the previous steps:

   For HDP:

   ```
   cb cluster maintenance-mode hdp --name "${clustername}" --cli-input-json
    ~/Users/myuser/stack.json
   ```

   For HDF:

   ```
   cb cluster maintenance-mode hdf --name "${clustername}" --cli-input-json
    ~/Users/myuser/stack.json
   ```

   Replace ~/Users/myuser/stack.json with the actual name and location of your JSON file.
5. (Optional) Verify that the update was successful by using cb cluster describe:

   ```
   cb cluster describe --name "${clustername}" | jq
    ".cluster.ambariStackDetails"
   ```

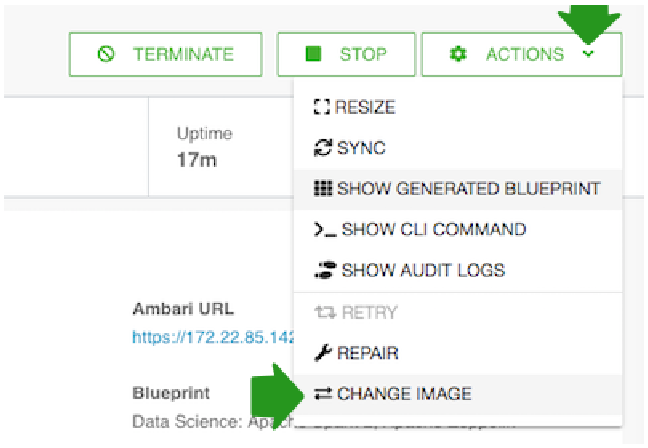
Step 5: Change cluster's default image

If using prewarmed images, the updated Ambari and/or stack version will be different from the original versions included in the prewarmed image, making it necessary to change prewarmed images. In this case, you should either update to a compatible prewarmed image (if one exist) or update to a base image. If you are using base images, skip this step.

Updating your cluster to use the new default image for newly added cluster nodes can be done by using the web UI or CLI.

(Option 1: web UI) To update cluster’s default image by using the web UI:

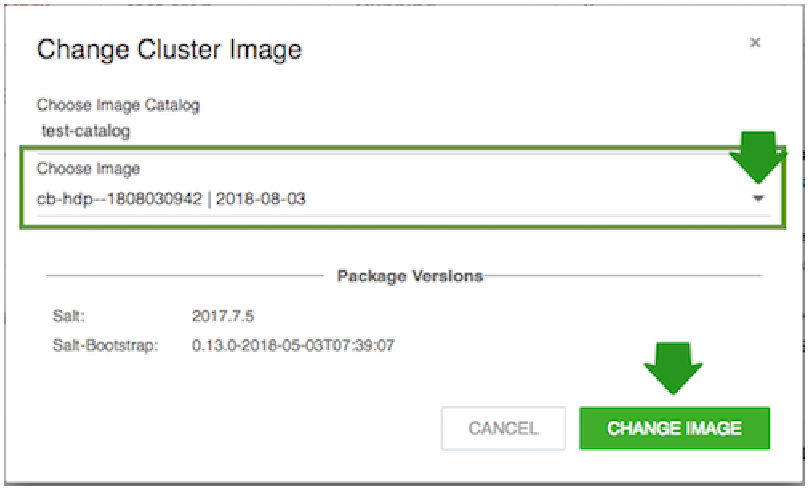
1. Navigate to the cluster details.
2. Sync the cluster (Actions > Sync) to ensure that Cloudbreak has the latest image and package information.
3. From the Actions menu select Change image.



4. Under Choose Image Catalog, choose the image catalog where the image that you would like to use is included.

Note:

If this window does not show any images, try



5. Under Choose Image, select the image that you would like to use as the new default image. Click on Choose image.

This will set the new image as default for all newly added nodes.

Note:

This will not update currently running cluster nodes. Only the nodes added after the default image update will use the new default image. Existing cluster nodes, must be updated separately.

(Option 2: CLI) To update cluster's default image by using the CLI:

1. Sync the cluster:

```
cb cluster sync --name "${clustername}"
```

2. List the available images you can upgrade your cluster to:

```
cb imagecatalog images cluster-upgrade --cluster "${clustername}"
```

3. Update your cluster's default image:

```
cb cluster change-image --name "${clustername}" --imageid <image-id>
```

Step 6: Validate repository configuration

Perform these steps to validate that the repository configurations are correct.

1. Sync the cluster (if not already done in the previous step):

```
cb cluster sync --name "${clustername}"
```

2. Run the following command to trigger validation:

```
cb cluster maintenance-mode validate --name ${clustername}
```

3. check the cluster's Event History in the web UI to ensure that the validation finished successfully. Upon successful validation, you will see the following messages:

```
Validation finished successfully.
The validation of the repo and image settings has begun
```

If you see any warnings, you should resolve them prior to disabling maintenance mode.

Step 7: Disable maintenance mode

Once you have successfully validated the repo configuration, disable maintenance mode by using the following CLI command:

```
cb cluster maintenance-mode disable --name "${clustername}"
```

Once you exit maintenance mode, all cluster operations will be available as usual.