Ambari 2

Managing High Availability

Date of Publish: 2019-08-26

https://docs.hortonworks.com

Contents

Managing high availability......3
- Enabling AMS high availability......3
- Configuring NameNode high availability......5
  - Enable NameNode high availability......5
  - Manage JournalNodes......12
  - Rolling back NameNode high availablity......16
- Configuring ResourceManager high availability......23
  - Enable ResourceManager high availability......23
  - Disable ResourceManager high availability......25
- Configuring HBase high availability......26
  - Add a new HBase master to an existing cluster......26
  - Add a standby HBase master to a new cluster......27
- Setting Up Multiple HBase Masters Manually......27
  - Configure passwordless ssh access for HBase......27
  - Prepare node-1 for Hbase......28
  - Prepare node-2 and node-3 for Hbase......28
  - Start and test your HBase cluster......28
- Configuring Hive high availability......30
  - Add a Hive metastore component......30
  - Add a HiveServer2 component......30
  - Add a WebHCat server......30
- Configuring Storm high availability......31
  - Add a Nimbus Component......31
- Configuring Oozie high availability......31
  - Add an Oozie server component......31
- Configuring Atlas high availability......32
- Enabling Ranger admin high availability......33

Managing high availability

Ambari web provides a wizard-driven user experience that enables you to configure high availability of the components that support services across the stack. High availability is assured through establishing active and standby components. If an active component becomes unavailable, the standby component is available. After configuring high availability for a service, Ambari enables you to manage and disable (roll back) high availability of components in that service.

Ambari web provides a wizard-driven user experience that enables you to configure high availability of the components that support services across the stack. High availability is assured through establishing active and standby components. In the event that the active component fails or becomes unavailable, the standby component is available. After configuring high availability for a service, Ambari enables you to manage and disable (roll back) high availability of components in that service.

Enabling AMS high availability

Make AMS highly available by switching AMS to distributed mode and adding a second, standby Metrics Collector.

About this task

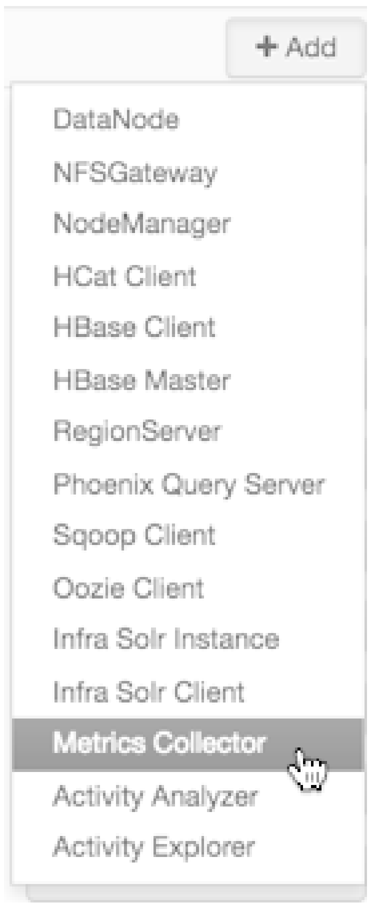
Ambari installs the Ambari Metrics System (AMS) , into the cluster with a single *Metrics Collector* component by default. The *Collector* is a daemon that runs on a specific host in the cluster and receives data from the registered publishers, the *Monitors* and *Sinks*. Depending on your needs, you might require AMS to have two collectors to cover a High Availability scenario. Establishing AMS High Availability means having a standby collector take over in the event the primary collector fails.

Before you begin

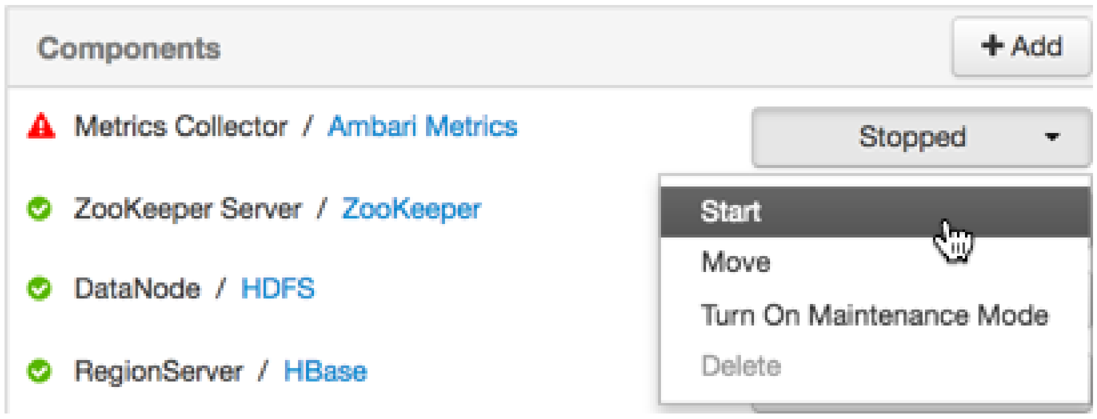
You must deploy AMS in distributed, not embedded mode.

Procedure

1. In **Ambari Web**, browse to the host where you would like to install another collector.
2. On the **Host** page, click **+Add**.



3. Click **Metrics Collector** from the **+Add** list.
   Ambari installs the new Metrics Collector and configures Ambari Metrics for HA. The new Collector will be installed in a stopped state.
4. In **Ambari Web** > **Components**, start the new Collector component.



If you attempt to add a second Collector to the cluster without first switching AMS to distributed mode, the collector will install but will not be able to be started. To resolve this:

1. Delete the newly added Collector.
2. Enable distributed mode.
3. Then, re-add the Collector.

Example

```
Traceback (most recent call last):
```

```
File
 "/var/lib/ambari-agent/cache/common-services/AMBARI_METRICS/0.1.0/package/
scripts/metrics_collector.py", line 150, in <module>
 AmsCollector().execute()
File
 "/usr/lib/python2.6/site-packages/resource_management/libraries/script/
script.py", line 313, in execute
 method(env)
File
"/var/lib/ambari-agent/cache/common-services/AMBARI_METRICS/0.1.0/package/
scripts/metrics_collector.py", line 48, in start self.configure(env, action
 = 'start') # for security
File
 "/usr/lib/python2.6/site-packages/resource_management/libraries/script/
script.py", line 116, in locking_configure original_configure(obj, *args,
 **kw)
File
 "/var/lib/ambari-agent/cache/common-services/AMBARI_METRICS/0.1.0/package/
scripts/metrics_collector.py", line 42, in configure raise
Fail("AMS in embedded mode cannot have more than 1 instance.
Delete all but 1 instances or switch to Distributed mode ")
 resource_management.core.exceptions.
Fail: AMS in embedded mode cannot have more than 1 instance.
Delete all but 1 instances or switch to Distributed mode
```

Configuring NameNode high availability

Enable NameNode high availability

To ensure that another NameNode in your cluster is always available if the active NameNode host fails, you should enable and configure NameNode high availability on your cluster using Ambari Web.

About this task

The **Enable HA wizard** describes the set of automated and manual steps you must take to set up NameNode high availability. HDFS and ZooKeeper must stop and start when enabling NameNode HA. Maintenance Mode will prevent those start and stop operations from occurring. If the HDFS or ZooKeeper services are in Maintenance Mode the NameNode HA wizard will not complete successfully.

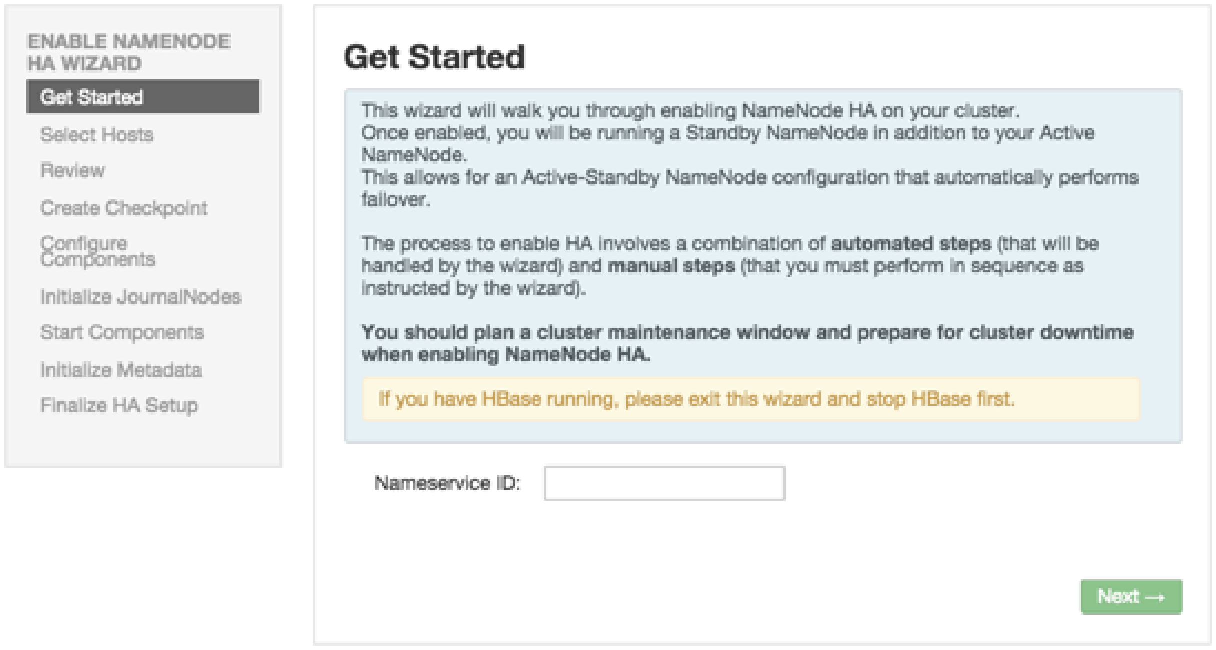
Before you begin

- Verify that you have at least three hosts in your cluster and are running at least three Apache ZooKeeper servers.
- Verify that the Hadoop Distributed File System (HDFS) and ZooKeeper services are not in Maintenance Mode.

Procedure

1. In **Ambari Web**, browse to **Services** > **HDFS** > **Summary**.
2. Click **Actions**, then click **Enable NameNode HA**.
   The Enable NameNode HA wizard launches.
3. On **Get Started**, type in a Nameservice ID and click **Next**.

   You use this Nameservice ID instead of the NameNode FQDN after HA is set up.



Get Started

This wizard will walk you through enabling NameNode HA on your cluster.
Once enabled, you will be running a Standby NameNode in addition to your Active NameNode.
This allows for an Active-Standby NameNode configuration that automatically performs failover.

The process to enable HA involves a combination of **automated steps** (that will be handled by the wizard) and **manual steps** (that you must perform in sequence as instructed by the wizard).

You should plan a cluster maintenance window and prepare for cluster downtime when enabling NameNode HA.

If you have HBase running, please exit this wizard and stop HBase first.

Nameservice ID:

If you are using Hive, you must manually change the Hive Metastore FS root to point to the Nameservice URI instead of the NameNode URI, as follows:

a. Find the current FS root on the Hive host.

hive --config /etc/hive/conf/conf.server --service metatool -listFSRoot

utput should look like: Listing FS Roots... hdfs://[NAMENODE_HOST]/apps/hive/warehouse.

b. Change the FS root.

$ hive --config /etc/hive/conf/conf.server --service metatool -updateLocation [NEW_LOCATION] [OLD_LOCATION]

For example, if your Nameservice ID is mycluster, you input:

$ hive --config /etc/hive/conf/conf.server --service metatool -updateLocation hdfs://mycluster/apps/hive/warehouse hdfs://c6401.ambari.apache.org/apps/hive/warehouse

The output looks similar to:

Successfully updated the following locations... Updated X records in SDS table

Important:

The Hive configuration path for a default HDP 2.3.x or later stack is /etc/hive/conf/conf.server

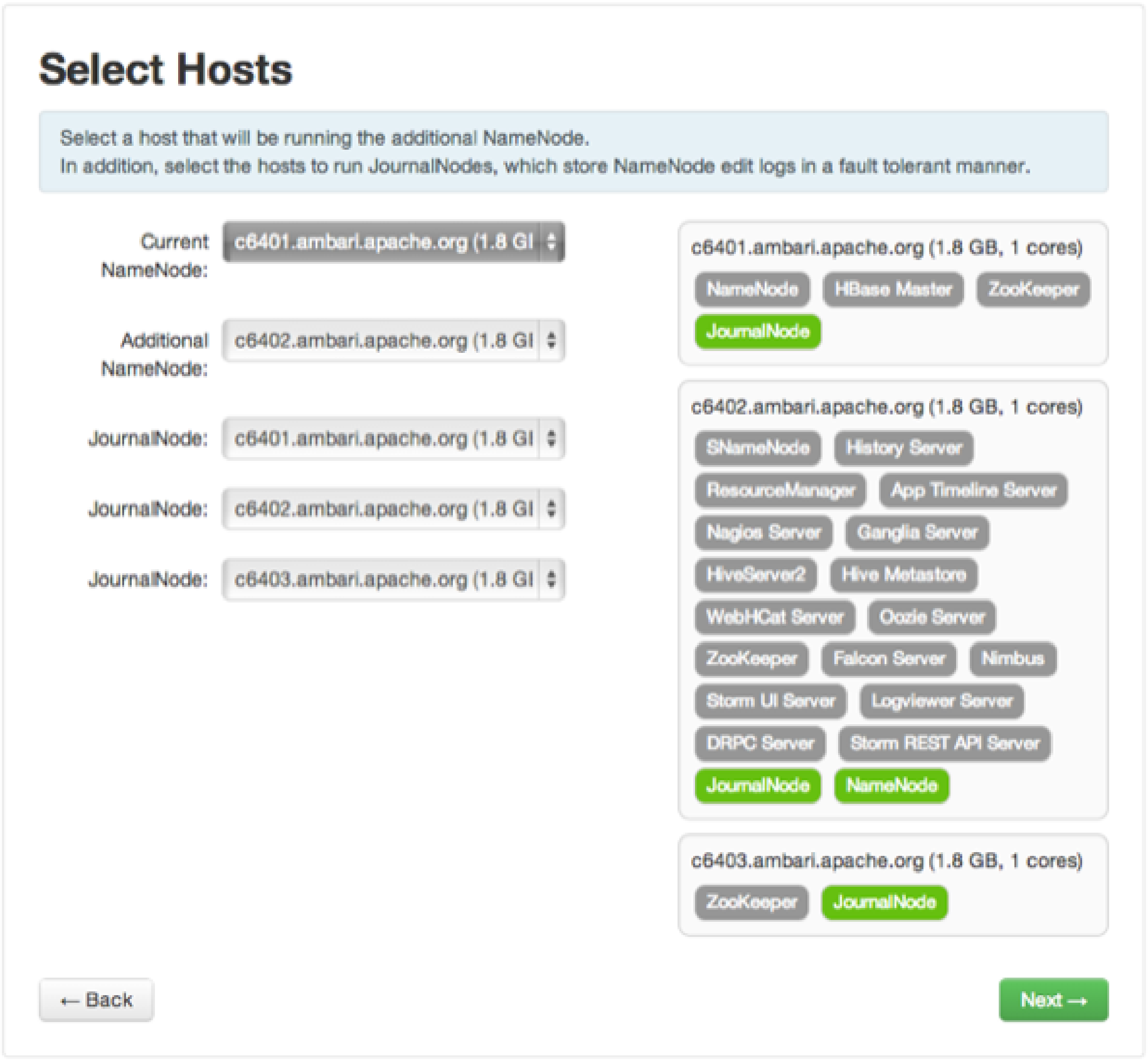
The Hive configuration path for a default HDP 2.2.x or earlier stack is /etc/hive/conf

c. Adjust the ZooKeeper Failover Controller retries setting for your environment.

1. Browse to **Services** > **HDFS** > **Configs** > **Advanced core-site**.

d. Set ha.failover-controller.active-standby-elector.zk.op.retries=120.

4. On **Select Hosts**, select a host for the additional NameNode and the JournalNodes, and then click **Next**.



Select Hosts

Select a host that will be running the additional NameNode.
In addition, select the hosts to run JournalNodes, which store NameNode edit logs in a fault tolerant manner.

Current NameNode: c6401.ambari.apache.org (1.8 GI

Additional NameNode: c6402.ambari.apache.org (1.8 GI

JournalNode: c6401.ambari.apache.org (1.8 GI

JournalNode: c6402.ambari.apache.org (1.8 GI

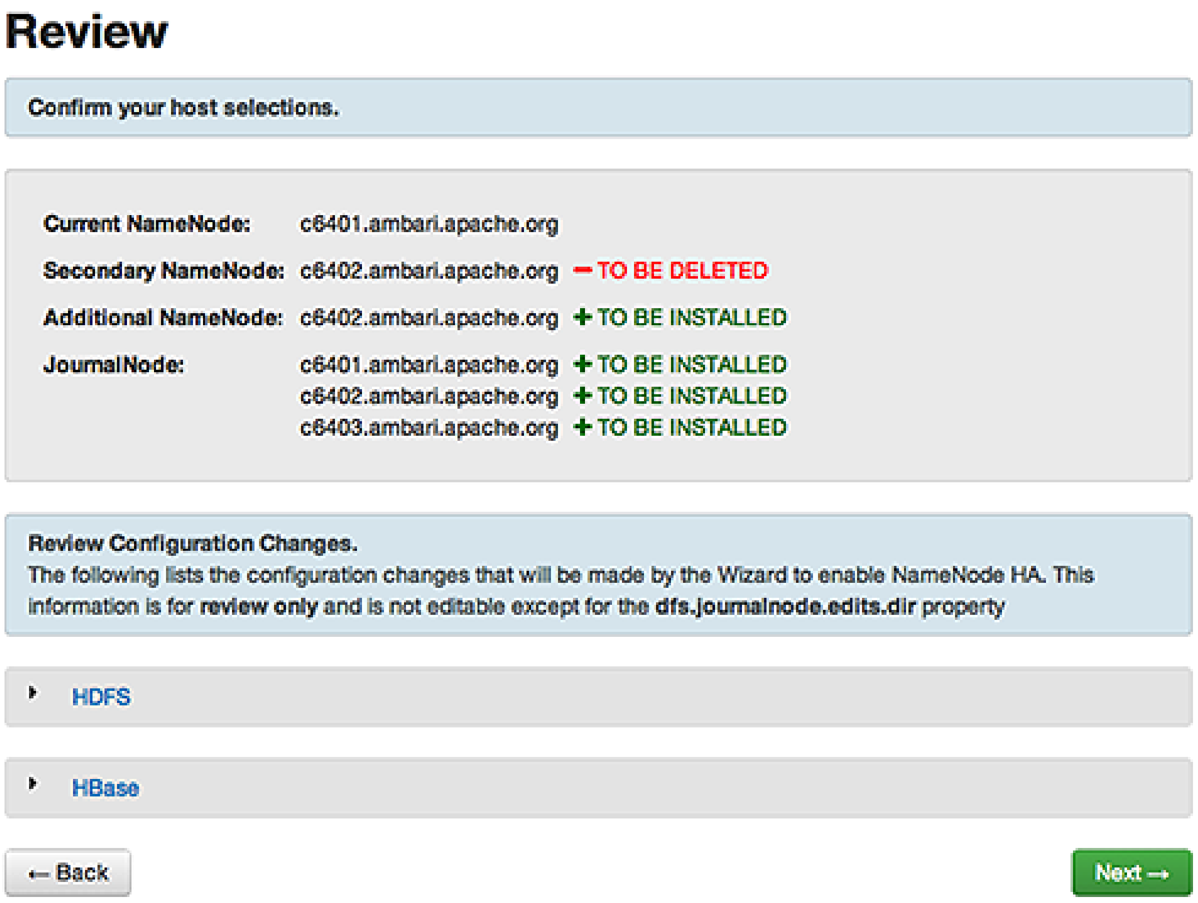
JournalNode: c6403.ambari.apache.org (1.8 GI

c6401.ambari.apache.org (1.8 GB, 1 cores)
NameNode, HBase Master, ZooKeeper, JournalNode

c6402.ambari.apache.org (1.8 GB, 1 cores)
SNameNode, History Server, ResourceManager, App Timeline Server, Nagios Server, Ganglia Server, HiveServer2, Hive Metastore, WebHCat Server, Oozie Server, ZooKeeper, Falcon Server, Nimbus, Storm UI Server, Logviewer Server, DRPC Server, Storm REST API Server, JournalNode, NameNode

c6403.ambari.apache.org (1.8 GB, 1 cores)
ZooKeeper, JournalNode

5. On Review, confirm your host selections and click **Next**:



Review

Confirm your host selections.

Current NameNode: c6401.ambari.apache.org

Secondary NameNode: c6402.ambari.apache.org — TO BE DELETED

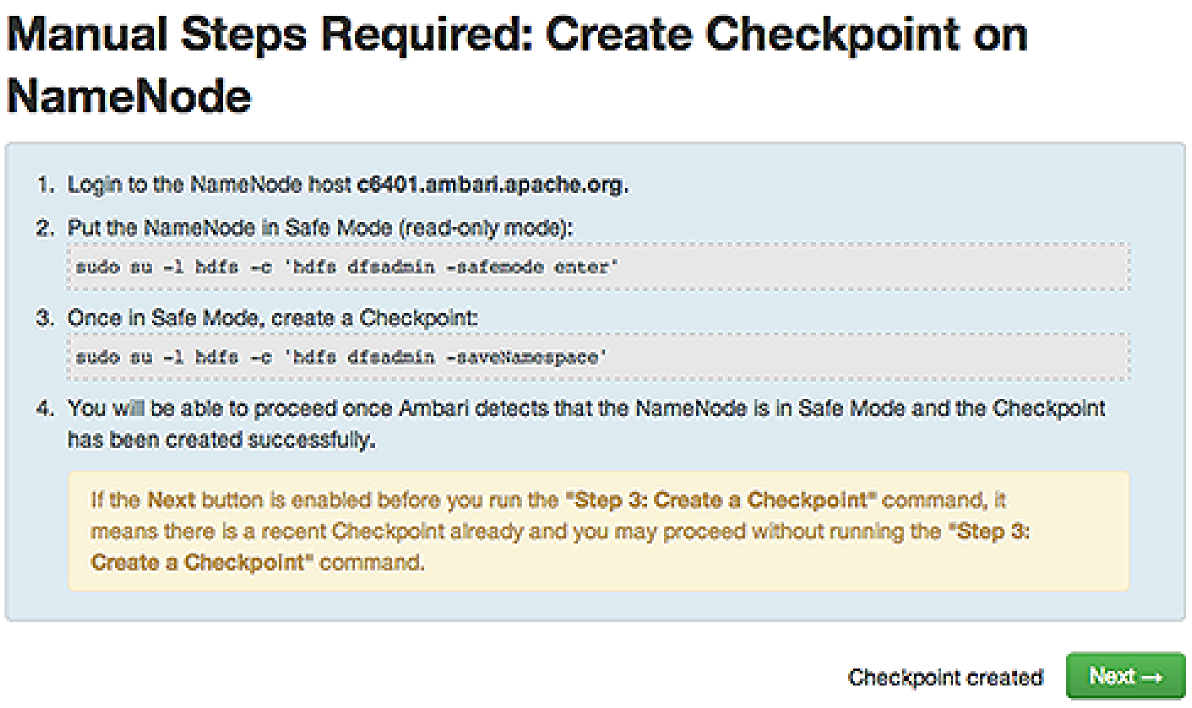
Additional NameNode: c6402.ambari.apache.org + TO BE INSTALLED

JournalNode: c6401.ambari.apache.org + TO BE INSTALLED
c6402.ambari.apache.org + TO BE INSTALLED
c6403.ambari.apache.org + TO BE INSTALLED

Review Configuration Changes.
The following lists the configuration changes that will be made by the Wizard to enable NameNode HA. This information is for **review only** and is not editable except for the **dfs.journalnode.edits.dir** property

6. Follow the directions on the **Manual Steps Required: Create Checkpoint on NameNode** page.

You must log in to your current NameNode host and run the commands to put your NameNode into safe mode and create a checkpoint.



Manual Steps Required: Create Checkpoint on NameNode

1. Login to the NameNode host **c6401.ambari.apache.org**.
2. Put the NameNode in Safe Mode (read-only mode):

```
sudo su -l hdfs -c 'hdfs dfsadmin -safemode enter'
```

3. Once in Safe Mode, create a Checkpoint:

```
sudo su -l hdfs -c 'hdfs dfsadmin -saveNamespace'
```

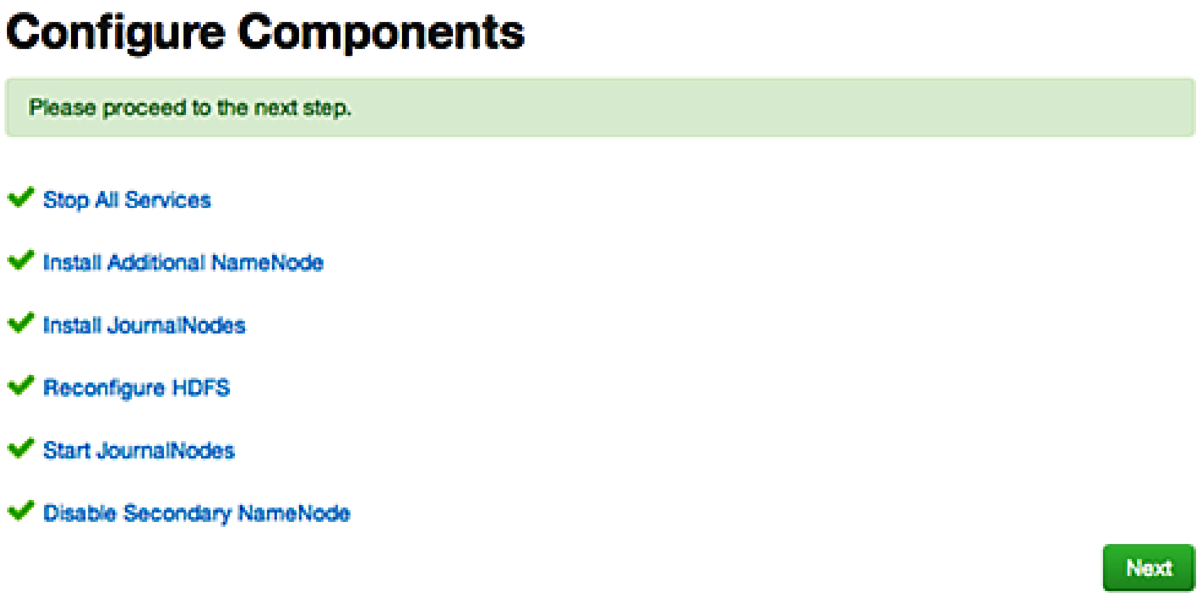
4. You will be able to proceed once Ambari detects that the NameNode is in Safe Mode and the Checkpoint has been created successfully.

If the **Next** button is enabled before you run the **"Step 3: Create a Checkpoint"** command, it means there is a recent Checkpoint already and you may proceed without running the **"Step 3: Create a Checkpoint"** command.

Checkpoint created

When Ambari detects success and the message on the bottom of the window changes to Checkpoint created, click **Next**.

7. On **Configure Components**, monitor the configuration progress bars.



Configure Components

Please proceed to the next step.

✓ Stop All Services

✓ Install Additional NameNode

✓ Install JournalNodes

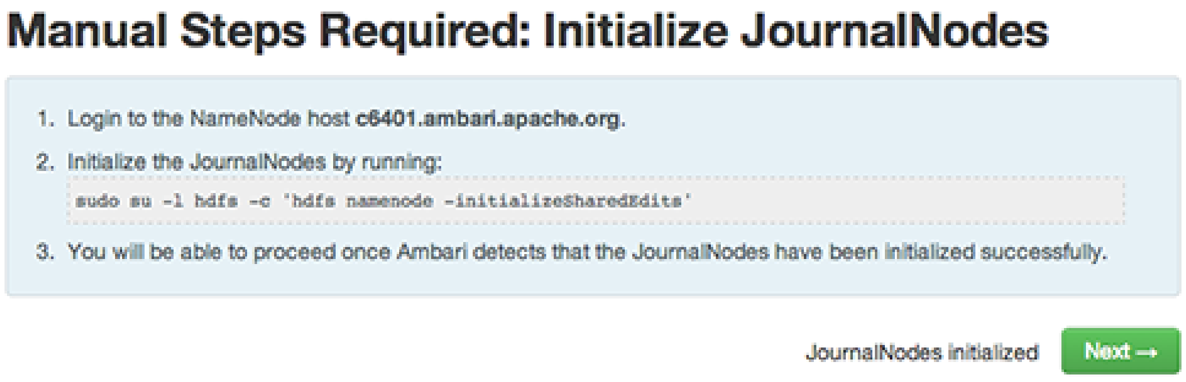
✓ Reconfigure HDFS

✓ Start JournalNodes

✓ Disable Secondary NameNode

When all componets have been configured successfully, click **Next**

8. Follow the instructions on **Manual Steps Required: Initialize JournalNodes**.



Manual Steps Required: Initialize JournalNodes

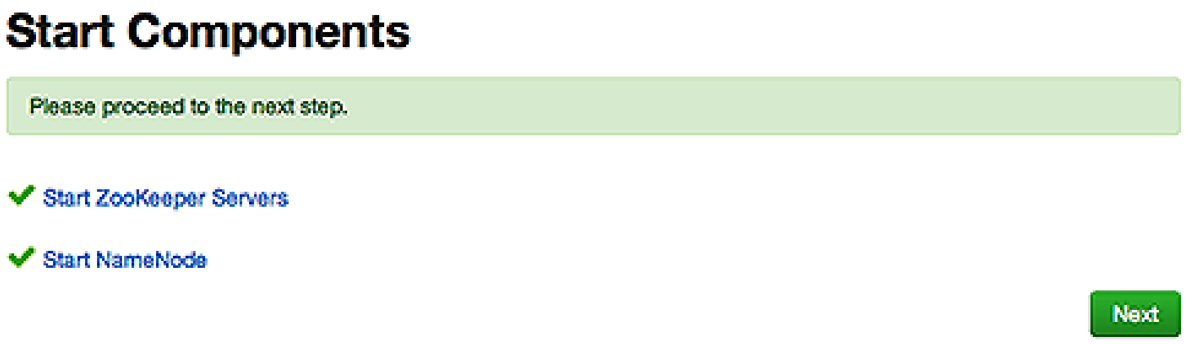
1. Login to the NameNode host **c6401.ambari.apache.org**.
2. Initialize the JournalNodes by running:

```
sudo su -l hdfs -c 'hdfs namenode -initializeSharedEdits'
```

3. You will be able to proceed once Ambari detects that the JournalNodes have been initialized successfully.

You must log in to your current NameNode host to run the command to initialize the JournalNodes. When JournalNodes initiate successfully, click **Next**.

9. On **Start Components**, monitor the progress bars as the ZooKeeper servers and NameNode start.



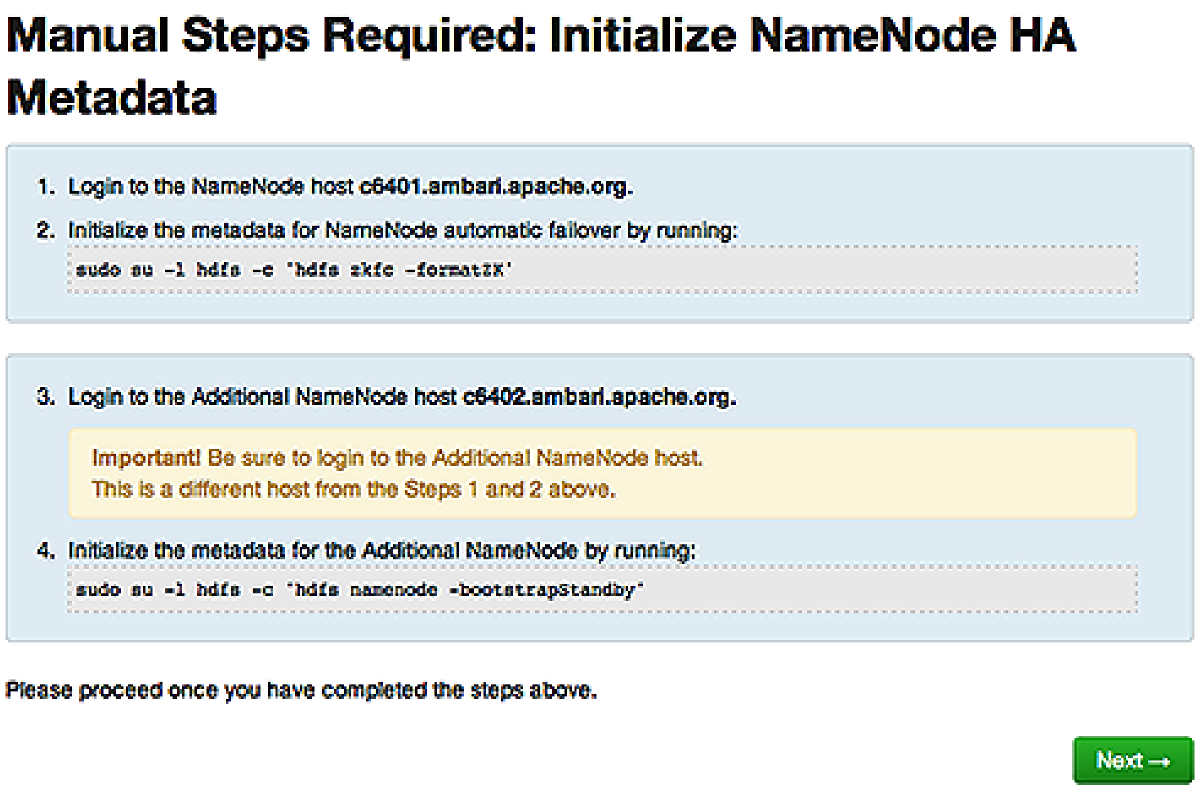
Start Components

Please proceed to the next step.

When ZooKeeper servers and NameNode have started, click **Next**.

In a cluster with Ranger enabled, and with Hive configured to use MySQL, Ranger will fail to start if MySQL is stopped. To work around this issue, start the Hive MySQL database and then retry starting components.

10. On **Manual Steps Required: Initialize NameNode HA Metadata**, complete all steps, using the instructions on the page.



Manual Steps Required: Initialize NameNode HA Metadata

1. Login to the NameNode host c6401.ambari.apache.org.
2. Initialize the metadata for NameNode automatic failover by running:

```
sudo su -l hdfs -c 'hdfs zkfc -formatZK'
```

3. Login to the Additional NameNode host c6402.ambari.apache.org.

Important! Be sure to login to the Additional NameNode host.
This is a different host from the Steps 1 and 2 above.

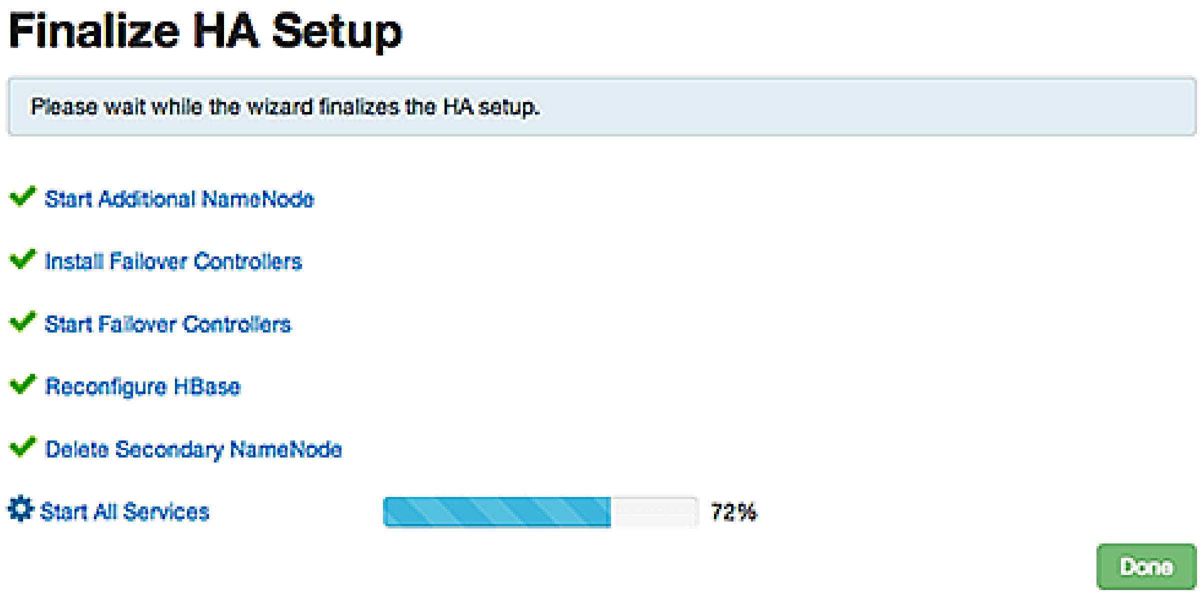
4. Initialize the metadata for the Additional NameNode by running:

```
sudo su -l hdfs -c 'hdfs namenode -bootstrapStandby'
```

Please proceed once you have completed the steps above.

For this step, you must log in to both the current NameNode and the additional NameNode. Make sure you are logged in to the correct host for each command. Click **OK** to confirm, after you complete each command. When you have completed the four steps required to intialize metadata, click **Next**.

11. On **Finalize HA Setup**, monitor the progress bars, the wizard completes HA setup.



Finalize HA Setup

Please wait while the wizard finalizes the HA setup.

✓ Start Additional NameNode
✓ Install Failover Controllers
✓ Start Failover Controllers
✓ Reconfigure HBase
✓ Delete Secondary NameNode
Start All Services
72%
Done

When the wizard completes HA setup, click **Done** to finish the wizard.

After **Ambari Web** reloads, you may see some alert notifications. Wait a few minutes until all the services restart.

12. Restart any components using Ambari Web, if necessary.

What to do next

Review and confirm all recommended configuration changes.

Related Information

Review and confirm configuration changes

Restart all required services

Manage JournalNodes

After enabling NameNode high availability, use the Manage JournalNodes wizards to maintain at least three active JournalNodes.

About this task

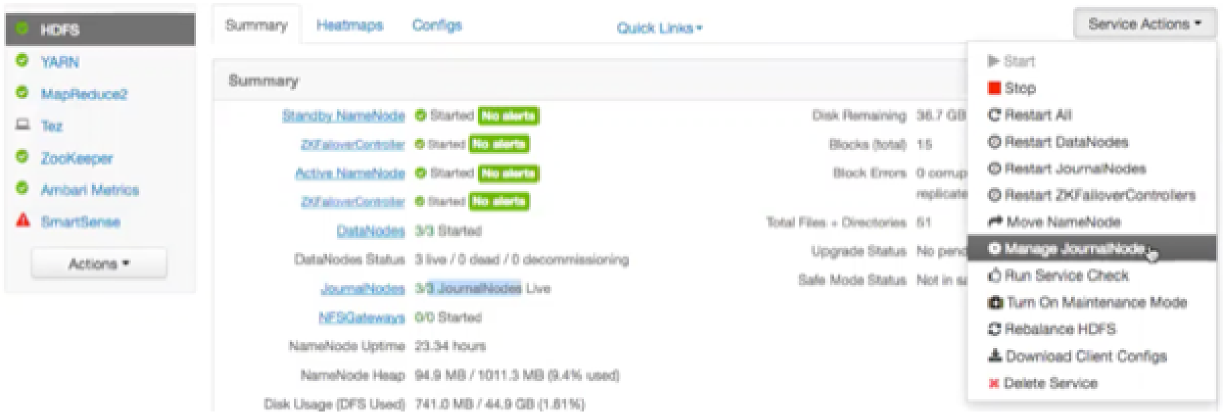
NameNode high availability requires that you must maintain at least three, active JournalNodes in your cluster. You can use the **Manage JournalNode** wizard to assign, add, or remove JournalNodes on hosts in your cluster. The **Manage JournalNode** wizard enables you to assign JournalNodes, review and confirm required configuration changes, and will restart all components in the cluster to take advantage of the changes made to JournalNode placement and configuration. Completing the **Manage JournalNode** wizard restarts all cluster services.

Before you begin

- NameNode high availability must be enabled in your cluster

Procedure

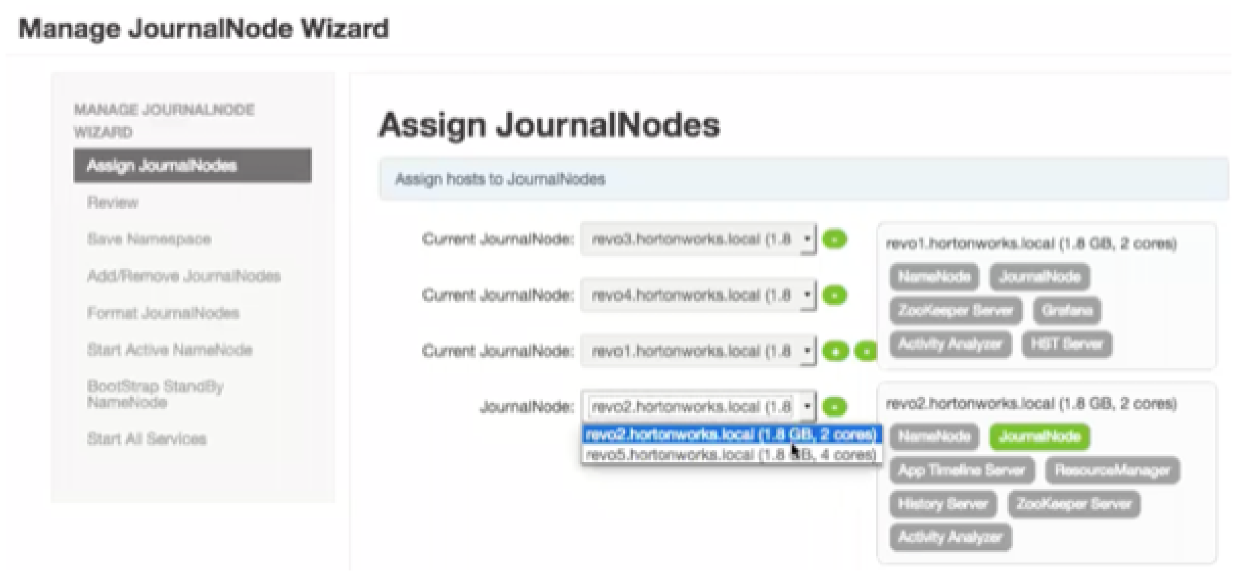
1. In **Ambari Web**, browse to **Services** > **HDFS** > **Summary**.
2. Click **Service Actions**, then click **Manage JournalNodes**.



Summary

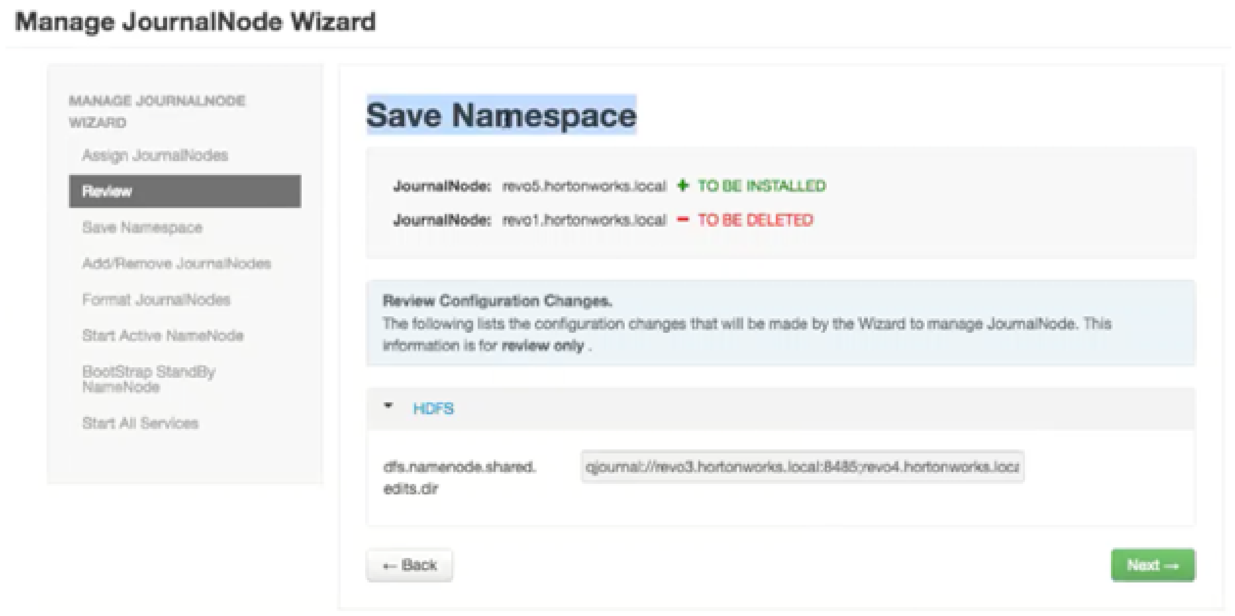
Standby NameNode Started No alerts
ZKFailoverController Started No alerts
Active NameNode Started No alerts
ZKFailoverController Started No alerts
DataNodes 3/3 Started
DataNodes Status 3 live / 0 dead / 0 decommissioning
JournalNodes 3/3 JournalNodes Live
NFSGateways 0/0 Started
NameNode Uptime 23.34 hours
NameNode Heap 94.9 MB / 1011.3 MB (9.4% used)
Disk Usage (DFS Used) 741.0 MB / 44.9 GB (1.61%)
Disk Remaining 36.7 GB
Blocks (total) 15
Block Errors 0 corrup replicate
Total Files + Directories 51
Upgrade Status No pend
Safe Mode Status Not in sa

3. On **Assign JournalNodes**, make assignments by clicking the **+** and **-** icons and selecting host names in the drop-down menus.


Manage JournalNode Wizard
MANAGE JOURNALNODE WIZARD
Assign JournalNodes
Review
Save Namespace
Add/Remove JournalNodes
Format JournalNodes
Start Active NameNode
BootStrap StandBy NameNode
Start All Services
Assign JournalNodes
Assign hosts to JournalNodes
Current JournalNode: revo3.hortonworks.local (1.8
Current JournalNode: revo4.hortonworks.local (1.8
Current JournalNode: revo1.hortonworks.local (1.8
JournalNode: revo2.hortonworks.local (1.8
revo2.hortonworks.local (1.8 GB, 2 cores)
revo5.hortonworks.local (1.8 GB, 4 cores)
revo1.hortonworks.local (1.8 GB, 2 cores)
NameNode
JournalNode
ZooKeeper Server
Grafana
Activity Analyzer
HST Server
revo2.hortonworks.local (1.8 GB, 2 cores)
NameNode
JournalNode
App Timeline Server
ResourceManager
History Server
ZooKeeper Server
Activity Analyzer

The **Assign JournalNodes** page enables you to maintain three, current JournalNodes by updating each time you make an assignment.

4. On **Review**, verify the summary of your JournalNode host assignments and the related configuration changes.



JournalNode: revo5.hortonworks.local + TO BE INSTALLED

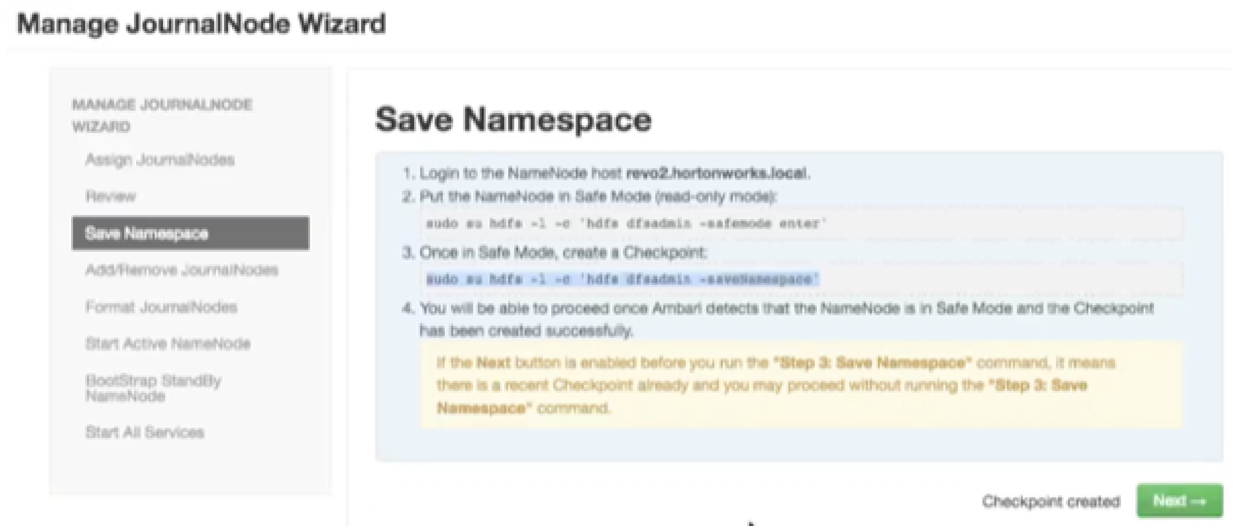
JournalNode: revo1.hortonworks.local − TO BE DELETED

Review Configuration Changes.
The following lists the configuration changes that will be made by the Wizard to manage JournalNode. This information is for **review only** .

dfs.namenode.shared.edits.dir qjournal://revo3.hortonworks.local:8485;revo4.hortonworks.loca

When you are satisfied that all assignments match your intentions, click **Next**.

5. Using a remote shell, complete the steps on **Save Namespace**.



Manage JournalNode Wizard

Save Namespace

1. Login to the NameNode host **revo2.hortonworks.local**.
2. Put the NameNode in Safe Mode (read-only mode):

```
sudo su hdfs -l -c 'hdfs dfsadmin -safemode enter'
```

3. Once in Safe Mode, create a Checkpoint:

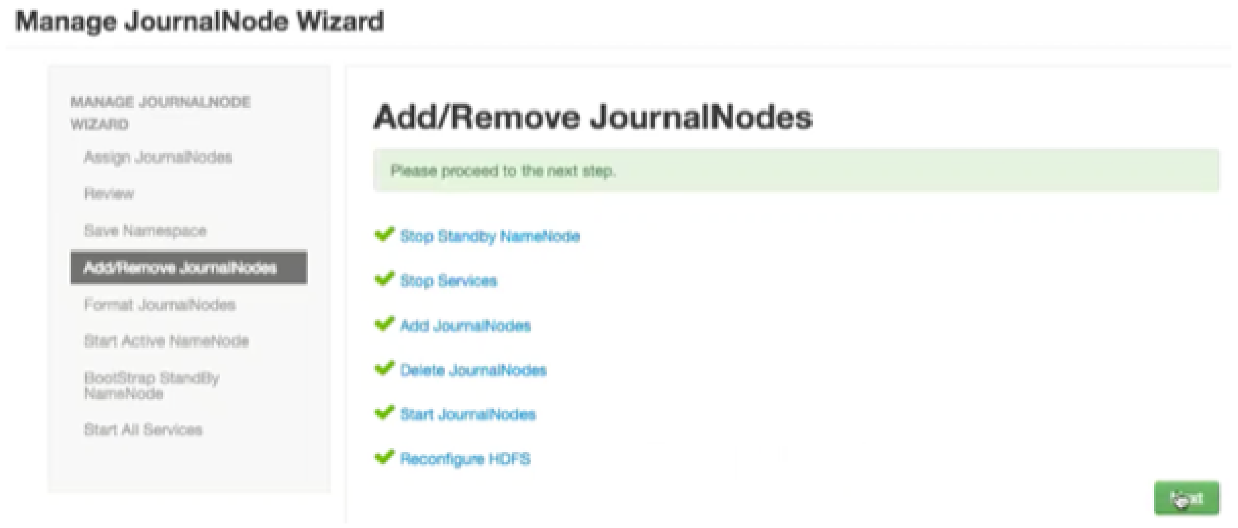
```
sudo su hdfs -l -c 'hdfs dfsadmin -saveNamespace'
```

4. You will be able to proceed once Ambari detects that the NameNode is in Safe Mode and the Checkpoint has been created successfully.

If the **Next** button is enabled before you run the **"Step 3: Save Namespace"** command, it means there is a recent Checkpoint already and you may proceed without running the **"Step 3: Save Namespace"** command.

When you have successfully created a checkpoint, click **Next**.

6. On **Add/Remove JournalNodes**, monitor the progress bars.



Manage JournalNode Wizard

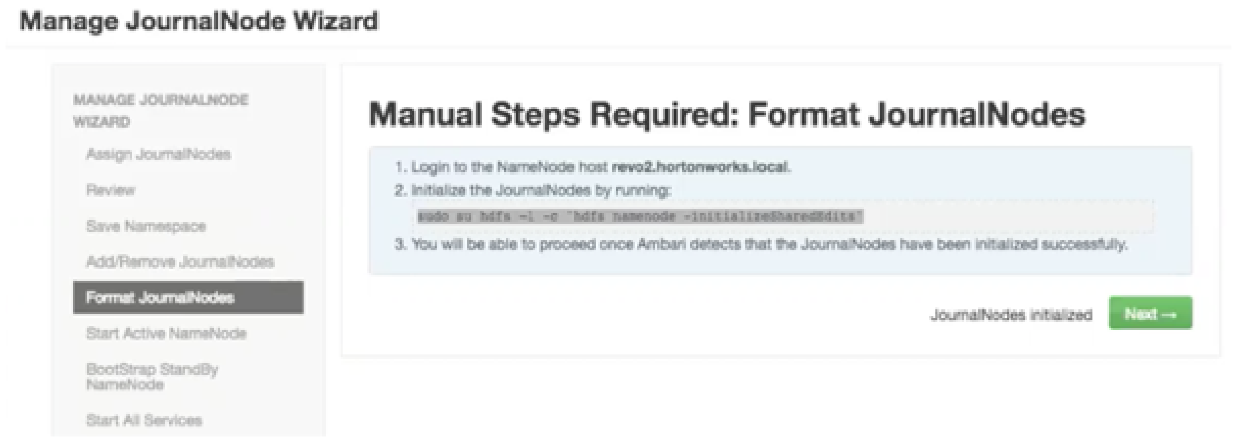
Add/Remove JournalNodes

Please proceed to the next step.

✓ Stop Standby NameNode
✓ Stop Services
✓ Add JournalNodes
✓ Delete JournalNodes
✓ Start JournalNodes
✓ Reconfigure HDFS

When progress completes, click **Next**.

7. Follow the instructions on **Manual Steps Required: Format JournalNodes**.



Manage JournalNode Wizard

Manual Steps Required: Format JournalNodes

1. Login to the NameNode host **revo2.hortonworks.local**.
2. Initialize the JournalNodes by running:

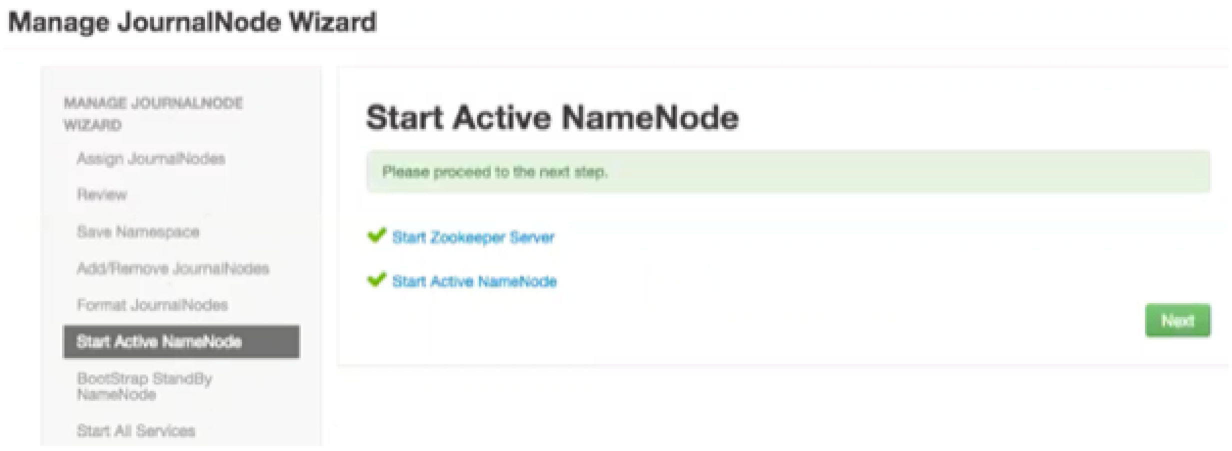
```
sudo su hdfs -l -c 'hdfs namenode -initializeSharedEdits'
```

3. You will be able to proceed once Ambari detects that the JournalNodes have been initialized successfully.

When JournalNodes have initialized, click **Next**.

8. In the remote shell, confirm that you want to initialize JournalNodes, by entering Y, at the following prompt: Re-format filesystem in QJM to [host.ip.address.1, host.ip.address.2, host.ip.address.3,] ? (Y or N) Y

9. On **Start Active NameNodes**, monitor the progress bars.



Start Active NameNode

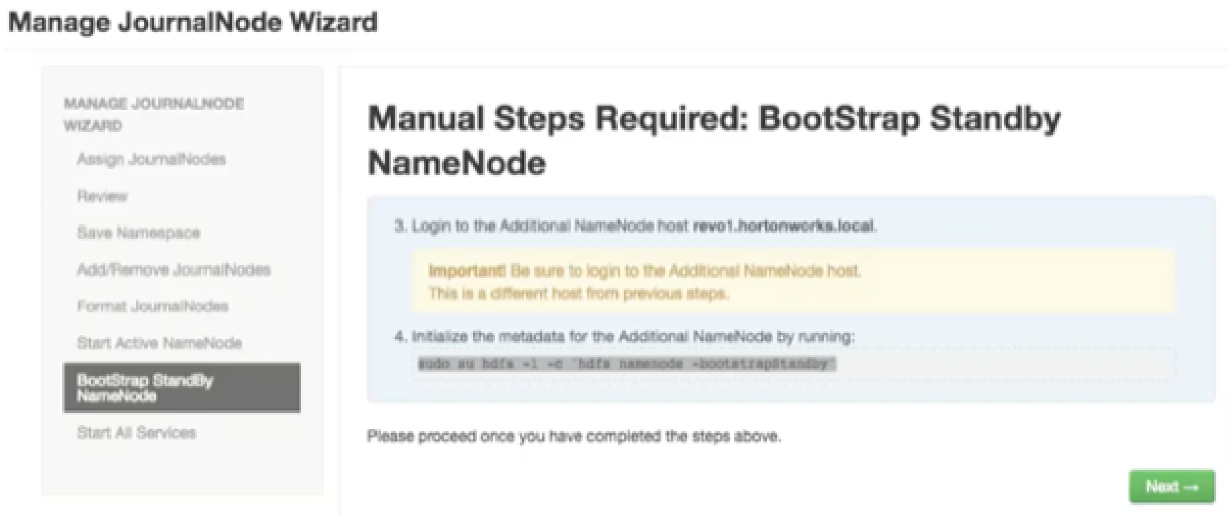
Please proceed to the next step.

✓ Start Zookeeper Server

✓ Start Active NameNode

When all services have re-started, click **Next**.

10. On **Manual Steps Required: Bootstrap Standby NameNode**, complete each step, using the instructions on the page.



Manual Steps Required: BootStrap Standby NameNode

3. Login to the Additional NameNode host **revo1.hortonworks.local**.

Important! Be sure to login to the Additional NameNode host.
This is a different host from previous steps.

4. Initialize the metadata for the Additional NameNode by running:

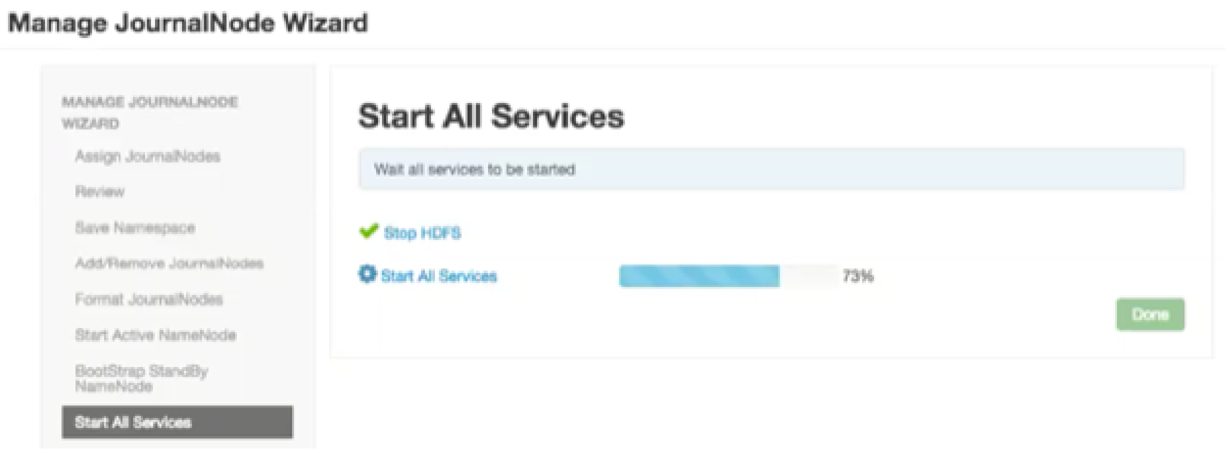
```
sudo su hdfs -l -c 'hdfs namenode -bootstrapStandby'
```

Please proceed once you have completed the steps above.

When you complete the bootstrap steps, click **Next**.

11. In the remote shell, confirm that you want to bootstrap the standby NameNode, by entering Y, at the following prompt:
RE-format filesystem in Storage Directory /grid/0/hadoop/hdfs/namenode ? (Y or N) Y

12. On **Start All Services**, monitor the progress bars.



Start All Services

Wait all services to be started

✓ Stop HDFS

Start All Services

When the wizard has started all services, click **Done**. After **Ambari Web** reloads, you may see some alert notifications. Wait a few minutes until all the services restart and alerts clear.

13. Restart any components using **Ambari Web**, if necessary.

What to do next

Review and confirm all recommended configuration changes.

Rolling back NameNode high availablity

To disable (roll back) NameNode high availability, perform these tasks (depending on your installation):

Stop HBase

Procedure

1. In **Ambari Web**, click the **HBase** service.
2. Click **Service Actions** > **Stop**.
3. Wait until HBase has stopped completely before continuing.

What to do next

Checkpoint the acitve NameNode.

Checkpoint the active NameNode

About this task

If HDFS is used after you enable NameNode HA, but you want to revert to a non-HA state, you must checkpoint the HDFS state before proceeding with the rollback. If the Enable NameNode HA wizard failed and you need to revert, you can omit this step and proceed to stop all services. Checkpointing the HDFS state requires different syntax, depending on whether Kerberos security is enabled on the cluster or not:

Procedure

- If Kerberos security has not been enabled on the cluster, use the following command on the active NameNode host and as the HDFS service user, to save the namespace:
  sudo su -l [HDFS_USER] -c 'hdfs dfsadmin -safemode enter' sudo su -l [HDFS_USER] -c 'hdfs dfsadmin -saveNamespace'
- If Kerberos security has been enabled on the cluster, use the following commands to save the namespace:
  sudo su -l [HDFS_USER] -c 'kinit -kt /etc/security/keytabs/nn.service.keytab nn/[HOSTNAME]@[REALM];hdfs dfsadmin -safemode enter' sudo su -l [HDFS_USER] -c 'kinit -kt /etc/security/keytabs/nn.service.keytab nn/[HOSTNAME]@[REALM];hdfs dfsadmin -saveNamespace'

  In this example, [HDFS_USER] is the HDFS service user (for example, hdfs), [HOSTNAME] is the Active NameNode hostname, and [REALM] is your Kerberos realm.

What to do next

Stop all services.

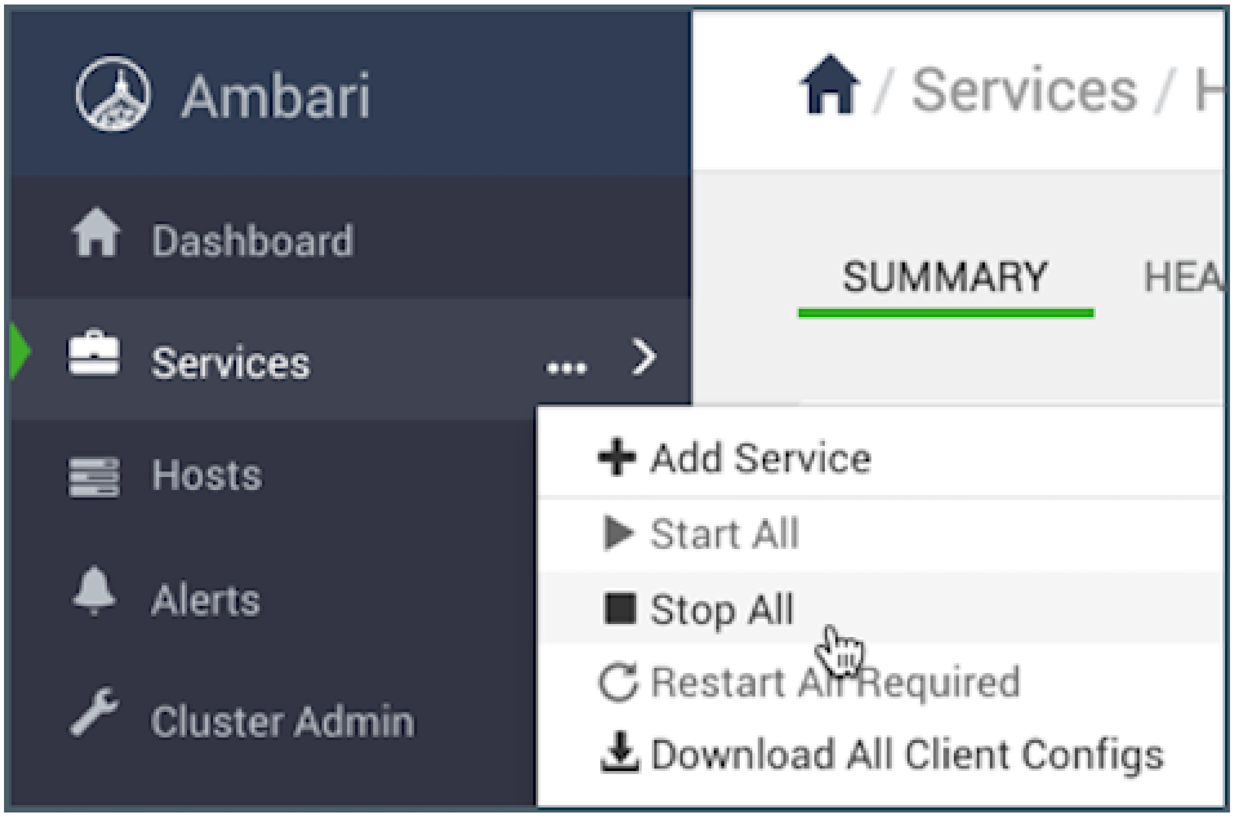
Stop all services

About this task

In **Ambari Web** > **Services** you can stop all listed services simultaneously.

Procedure

- In **Services**, click **...** and then click **Stop All**.



All listed services stop.

What to do next

Prepare the Ambari server host for NameNode rollback.

Prepare the Ambari server host for NameNode rollback

To prepare for the NameNode rollback task:

Procedure

1. Log in to the Ambari server host.
2. Set the following environment variables:

export AMBARI_USER=AMBARI_USERNAME	Substitute the value of the administrative user for Ambari Web. The default value is admin.
export AMBARI_PW=AMBARI_PASSWORD	Substitute the value of the administrative password for Ambari Web. The default value is admin.
export AMBARI_PORT=AMBARI_PORT	Substitute the Ambari Web port. The default value is 8080.
export AMBARI_PROTO=AMBARI_PROTOCOL	Substitute the value of the protocol for connecting to Ambari Web. Options are http or https. The default value is http.
export CLUSTER_NAME=CLUSTER_NAME	Substitute the name of your cluster, which you set during installation: for example, mycluster.
export NAMENODE_HOSTNAME=NN_HOSTNAME	Substitute the FQDN of the host for the non-HA NameNode: for example, nn01.mycompany.com.

export ADDITIONAL_NAMENODE_HOSTNAME=ANN_HOSTNAME	Substitute the FQDN of the host for the additional NameNode in your HA setup.
export SECONDARY_NAMENODE_HOSTNAME=SNN_HOSTNAME	Substitute the FQDN of the host for the standby NameNode for the non-HA setup.
export JOURNALNODE1_HOSTNAME=JOUR1_HOSTNAME	Substitute the FQDN of the host for the first Journal Node.
export JOURNALNODE2_HOSTNAME=JOUR2_HOSTNAME	Substitute the FQDN of the host for the second Journal Node.
export JOURNALNODE3_HOSTNAME=JOUR3_HOSTNAME	Substitute the FQDN of the host for the third Journal Node.

3. Double check that these environment variables are set correctly.

What to do next

Restore the HBase configuration.

Restore the HBase configuration

About this task

If you have installed HBase, you might need to restore a configuration to its pre-HA state:

Procedure

1. From the Ambari server host, determine whether your current HBase configuration must be restored:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] get localhost [CLUSTER_NAME] hbase-site

   Use the environment variables that you set up when preparing the Ambari server host for rollback for the variable names.

   - If hbase.rootdir is set to the NameService ID you set up using the **Enable NameNode HA** wizard, you must revert hbase-site to non-HA values. For example, in "hbase.rootdir":"hdfs://[name-service-id]:8020/apps/hbase/data", the hbase.rootdir property points to the NameService ID and the value must be rolled back.
   - If hbase.rootdir points instead to a specific NameNode host, it does not need to be rolled back. For example, in "hbase.rootdir":"hdfs://[nn01.mycompany.com]:8020/apps/hbase/data", the hbase.rootdir property points to a specific NameNode host and not a NameService ID. This does not need to be rolled back; you can proceed to delete ZooKeeper failover controllers.
2. If you must roll back the hbase.rootdir value, on the Ambari Server host, use the config.sh script to make the necessary change:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p[AMBARI_PW] -port [AMBARI_PORT] set localhost [CLUSTER_NAME] hbase-site hbase.rootdir hdfs://[NAMENODE_HOSTNAME]:8020/apps/hbase/data

   Use the environment variables that you set up when preparing the Ambari server host for rollback for the variable names.

   For Ambari 2.6.0 and higher, config.sh is not supported and will fail. Use config.py instead.
3. On the Ambari server host, verify that the hbase.rootdir property has been restored properly:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] get localhost [CLUSTER_NAME] hbase-site

   The hbase.rootdir property should now be the same as the NameNode hostname, not the NameService ID.

What to do next

Delete the ZooKeeper failover controllers.

Delete ZooKeeper failover controllers

Before you begin

If the following command on the Ambari server host returns an empty items array then you must delete ZooKeeper (ZK) Failover Controllers:

curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=ZKFC

Procedure

1. On the Ambari Server host, issue the following DELETE commands:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X DELETE [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts/[NAMENODE_HOSTNAME]/host_components/ZKFC curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X DELETE [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts/[ADDITIONAL_NAMENODE_HOSTNAME]/host_components/ZKFC
2. Verify that the controllers are gone:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=ZKFC

   This command should return an empty items array.

What to do next

Modify HDFS configurations.

Modify HDFS configurations

You may need to modify your hdfs-site configuration and/or your core-site configuration.

Before you begin

Check whether you need to modify your hdfs-site configuration, by executing the following command on the Ambari Server host:

/var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] get localhost [CLUSTER_NAME] hdfs-site

If you see any of the following properties, you must delete them from your configuration.

- dfs.nameservices
- dfs.client.failover.proxy.provider.[NAMESERVICE_ID]
- dfs.ha.namenodes.[NAMESERVICE_ID]
- dfs.ha.fencing.methods
- dfs.ha.automatic-failover.enabled
- dfs.namenode.http-address.[NAMESERVICE_ID].nn1
- dfs.namenode.http-address.[NAMESERVICE_ID].nn2
- dfs.namenode.rpc-address.[NAMESERVICE_ID].nn1
- dfs.namenode.rpc-address.[NAMESERVICE_ID].nn2
- dfs.namenode.shared.edits.dir
- dfs.journalnode.edits.dir
- dfs.journalnode.http-address
- dfs.journalnode.kerberos.internal.spnego.principal
- dfs.journalnode.kerberos.principal
- dfs.journalnode.keytab.file

Where [NAMESERVICE_ID] is the NameService ID you created when you ran the **Enable NameNode HA** wizard.

Procedure

1. Note -

For Ambari 2.6.0 and higher, config.sh is not supported and will fail. Use config.py instead.

2. On the Ambari Server host, execute the following for each property you found:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] delete localhost [CLUSTER_NAME] hdfs-site [PROPERTY_NAME]

   Replace [PROPERTY_NAME] with the name of each of the properties to be deleted.
3. Verify that all of the properties have been deleted:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] get localhost [CLUSTER_NAME] hdfs-site

   None of the properties listed above should be present.
4. Determine whether you must modify your core-site configuration:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] get localhost [CLUSTER_NAME] core-site
5. If you see the property ha.zookeeper.quorum, delete it:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] delete localhost [CLUSTER_NAME] core-site ha.zookeeper.quorum
6. If the property fs.defaultFS is set to the NameService ID, revert it to its non-HA value:
   "fs.defaultFS":"hdfs://[name-service-id]" The property fs.defaultFS needs to be modified as it points to a NameService ID "fs.defaultFS":"hdfs://[nn01.mycompany.com]"

   You need not change the property fs.defaultFS, because it points to a specific NameNode, not to a NameService ID.
7. Revert the property fs.defaultFS to the NameNode host value:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] set localhost [CLUSTER_NAME] core-site fs.defaultFS hdfs://[NAMENODE_HOSTNAME]
8. Verify that the core-site properties are now properly set:
   /var/lib/ambari-server/resources/scripts/configs.py -u [AMBARI_USER] -p [AMBARI_PW] -port [AMBARI_PORT] get localhost [CLUSTER_NAME] core-site

   The property fs.defaultFS should be the NameNode host and the property ha.zookeeper.quorum should not appear.

What to do next

Re-create the standby NameNode.

Re-create the standby NameNode

You may need to recreate your standby NameNode.

Before you begin

Check whether you need to recreate the standby NameNode, on the Ambari Server host:

curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X GET [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=SECONDARY_NAMENODE

If this returns an empty items array, you must recreate your standby NameNode. Otherwise you can proceed to re-enable your standby NameNode.

Procedure

1. On the Ambari Server host, run the following command:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X POST -d '{"host_components" : [{"HostRoles":{"component_name":"SECONDARY_NAMENODE"}}] }' [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts?Hosts/host_name=[SECONDARY_NAMENODE_HOSTNAME]
2. Verify that the standby NameNode now exists. On the Ambari Server host, run the following command:

curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X GET [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=SECONDARY_NAMENODE

This should return a non-empty items array containing the standby NameNode.

What to do next

Re-enable the standby NameNode.

Re-enable the standby NameNode

To re-enable the standby NameNode:

Procedure

1. Run the following commands on the Ambari Server host:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X PUT -d '{"RequestInfo":{"context":"Enable Secondary NameNode"},"Body":{"HostRoles":{"state":"INSTALLED"}}}'[AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts/[SECONDARY_NAMENODE_HOSTNAME}/host_components/SECONDARY_NAMENODE
2. Analyze the output.
3. If this returns 200, proceed to delete all JournalNodes.
4. If this input returns the value 202, wait a few minutes and then run the following command:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X GET "[AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=SECONDARY_NAMENODE&fields=HostRoles/state"
5. Wait for the response "state" : "INSTALLED" before proceeding.

What to do next

Delete all JournalNodes.

Delete all JournalNodes

You may need to delete any JournalNodes.

Before you begin

Check to see if you need to delete JournalNodes, on the Ambari Server host:

curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X GET [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=JOURNALNODE

If this returns an empty items array, you can go on to delete the additional NameNode. Otherwise you must delete the JournalNodes.

Procedure

1. On the Ambari Server host, run the following command:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X DELETE [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts/[JOURNALNODE1_HOSTNAME]/host_components/JOURNALNODE curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X DELETE [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts/[JOURNALNODE2_HOSTNAME]/host_components/JOURNALNODE
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X DELETE [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts/[JOURNALNODE3_HOSTNAME]/host_components/JOURNALNODE
2. Verify that all the JournalNodes have been deleted. On the Ambari Server host:

curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X GET [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=JOURNALNODE

This should return an empty items array.

What to do next

Delete all additional NameNodes.

Delete the additional NameNode

You may need to delete your additional NameNode.

Before you begin

Check to see if you need to delete your Additional NameNode, on the Ambari Server host:

curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X GET [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=NAMENODE

If the items array contains two NameNodes, the Additional NameNode must be deleted.

Procedure

1. On the Ambari Server host, run the following command:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X DELETE [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/hosts/[ADDITIONAL_NAMENODE_HOSTNAME]/host_components/NAMENODE
2. Verify that the Additional NameNode has been deleted:
   curl -u [AMBARI_USER]:[AMBARI_PW] -H "X-Requested-By: ambari" -i -X GET [AMBARI_PROTO]://localhost:[AMBARI_PORT]/api/v1/clusters/[CLUSTER_NAME]/host_components?HostRoles/component_name=NAMENODE

   This should return an items array that shows only one NameNode.

What to do next

Verify HDFS components.

Verify the HDFS components

Procedure

1. Browse to **Ambari Web UI** > **Services** > **HDFS**.
2. Check the Summary panel and ensure that the first three lines look like this:

   - NameNode
   - SNameNode
   - DataNodes

   You should not see a line for JournalNodes.

What to do next

Start HDFS service.

Start HDFS

Before you begin

Before starting HDFS, verify that you have the correct components.

Procedure

1. In **Ambari Web**, click **Service Actions**, then click **Start**.
2. If the progress bar does not show that the service has completely started and has passed the service checks, repeat Step 1.
3. To start all of the other services, click **Actions** > **Start All** in the **Services** navigation panel.

Configuring ResourceManager high availability

You can configure high availability for ResourceManager by using the Enable ResourceManager HA wizard.

Enable ResourceManager high availability

To access the wizard and enable ResourceManager high availability:

About this task

If you are working in an HDP 2.2 or later environment, you can configure high availability for ResourceManager by using the Enable ResourceManager HA wizard.

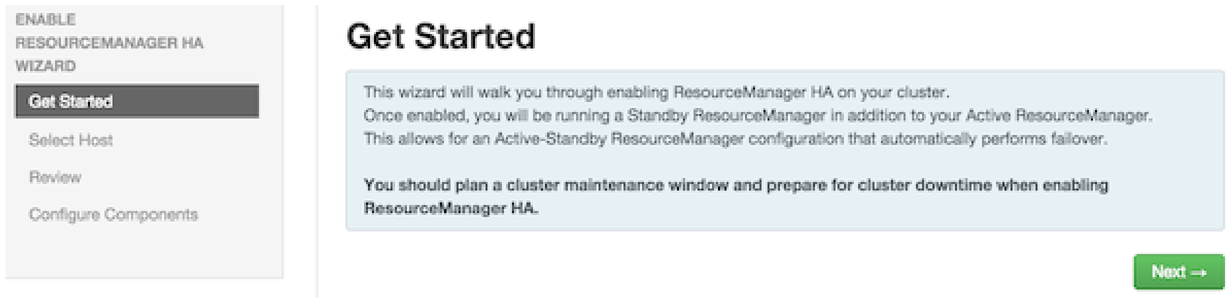
Before you begin

You must have at least three:

- hosts in your cluster
- Apache ZooKeeper servers running

Procedure

1. In **Ambari Web**, browse to **Services** > **YARN** > **Summary**.
2. Select Service Actions and choose Enable ResourceManager HA.
   The **Enable ResourceManager HA** wizard launches, describing a set of automated and manual steps that you must take to set up ResourceManager high availability.
3. On **Get Started**, read the overview of enabling ResourceManager HA.

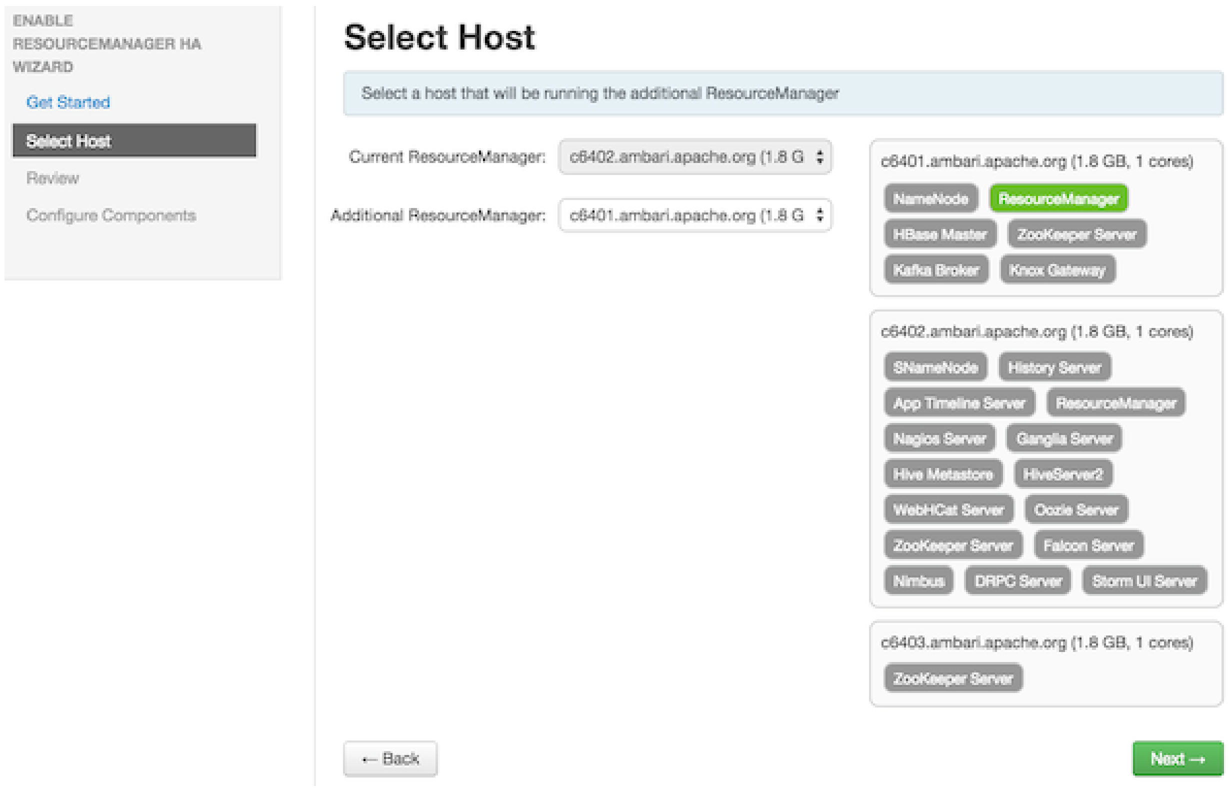
   

   Get Started

   This wizard will walk you through enabling ResourceManager HA on your cluster.
   Once enabled, you will be running a Standby ResourceManager in addition to your Active ResourceManager.
   This allows for an Active-Standby ResourceManager configuration that automatically performs failover.

   You should plan a cluster maintenance window and prepare for cluster downtime when enabling ResourceManager HA.

   Click **Next** to proceed.
4. On **Select Host**, accept the default selection or choose an available host.



Select Host

Select a host that will be running the additional ResourceManager

Current ResourceManager: c6402.ambari.apache.org (1.8 G

Additional ResourceManager: c6401.ambari.apache.org (1.8 G

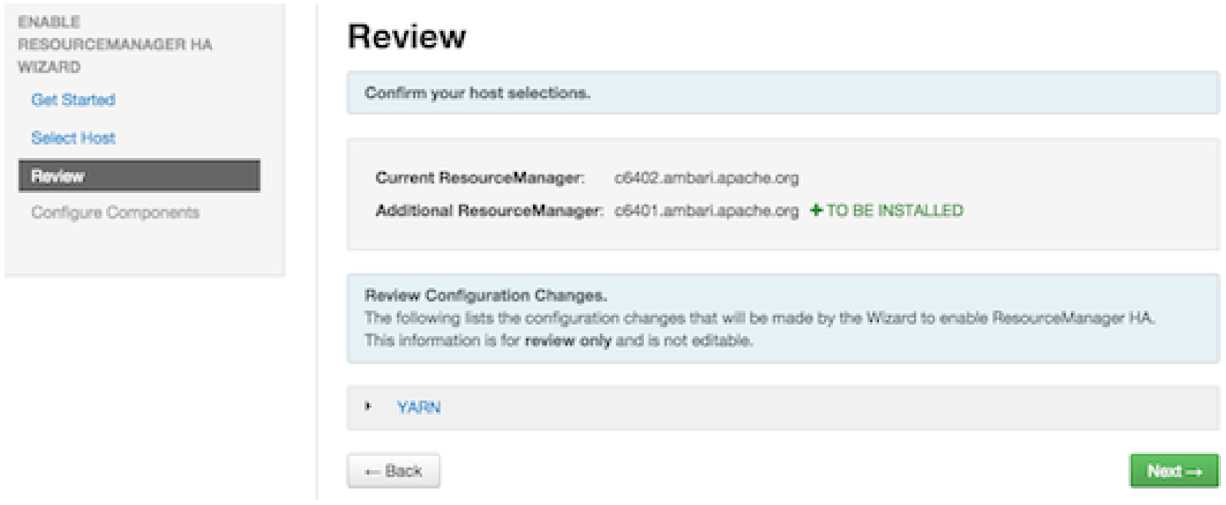
c6401.ambari.apache.org (1.8 GB, 1 cores)
NameNode, ResourceManager, HBase Master, ZooKeeper Server, Kafka Broker, Knox Gateway

c6402.ambari.apache.org (1.8 GB, 1 cores)
SNameNode, History Server, App Timeline Server, ResourceManager, Nagios Server, Ganglia Server, Hive Metastore, HiveServer2, WebHCat Server, Oozie Server, ZooKeeper Server, Falcon Server, Nimbus, DRPC Server, Storm UI Server

c6403.ambari.apache.org (1.8 GB, 1 cores)
ZooKeeper Server

Click **Next** to proceed.

5. On **Review Selections**, expand YARN if necessary, to review all the configuration changes proposed for YARN.



Review

Confirm your host selections.

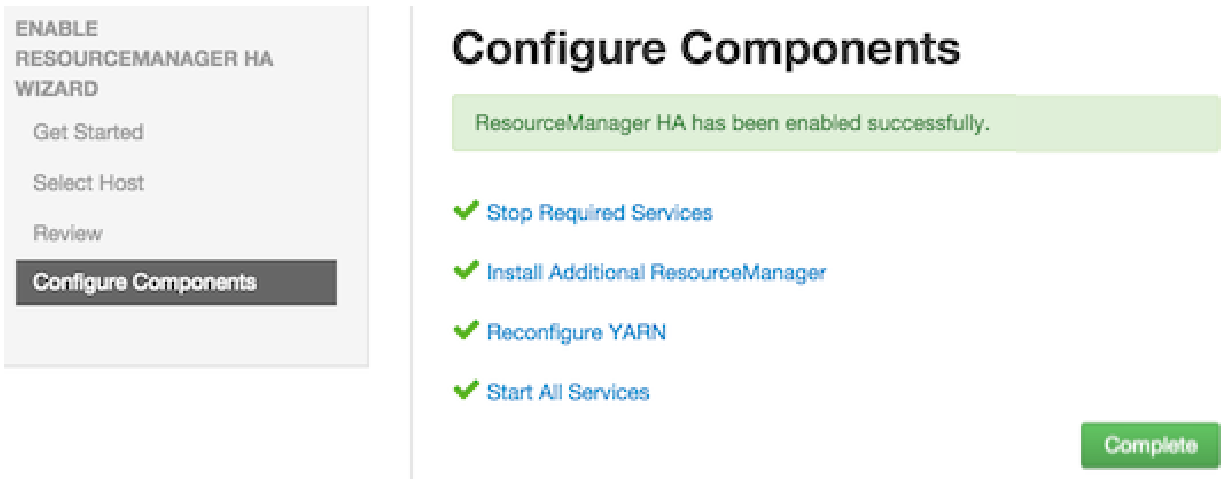
Current ResourceManager: c6402.ambari.apache.org

Additional ResourceManager: c6401.ambari.apache.org + TO BE INSTALLED

Review Configuration Changes.
The following lists the configuration changes that will be made by the Wizard to enable ResourceManager HA.
This information is for **review only** and is not editable.

Click **Next** to approve the changes and start automatically configuring ResourceManager HA.

6. On **Configure Components**, click **Complete** when all the progress bars finish tracking.



Configure Components

ResourceManager HA has been enabled successfully.

✓ Stop Required Services

✓ Install Additional ResourceManager

✓ Reconfigure YARN

✓ Start All Services

Disable ResourceManager high availability

Using the Ambari API, delete the ResourceManager. Then, use the ZooKeeper client to update the znode permissions.

About this task

To disable ResourceManager high availability, you must delete one ResourceManager and keep one ResourceManager. This requires using the Ambari API to modify the cluster configuration to delete the ResourceManager and using the ZooKeeper client to update the znode permissions.

Before you begin

Because these steps involve using the Ambari REST API, you should test and verify them in a test environment prior to executing against a production environment.

Procedure

1. In **Ambari Web**, stop YARN and ZooKeeper services.
2. On the Ambari server host, use the Ambari API to retrieve the YARN configurations into a JSON file.
   /var/lib/ambari-server/resources/scripts/configs.py get [AMBARI_SERVER] [CLUSTER_NAME] yarn-site yarn-site.json

   In this example, [AMBARI_SERVER] is the hostname of your Ambari Server and [CLUSTER_NAME] is the name of your cluster.

   For Ambari 2.6.0 and higher, config.sh is not supported and will fail. Use config.py instead.
3. In the yarn-site.json file, change yarn.resourcemanager.ha.enabled to false and delete the following properties:
   - yarn.resourcemanager.ha.rm-ids
   - yarn.resourcemanager.hostname.rm1
   - yarn.resourcemanager.hostname.rm2
   - yarn.resourcemanager.webapp.address.rm1
   - yarn.resourcemanager.webapp.address.rm2
   - yarn.resourcemanager.webapp.https.address.rm1
   - yarn.resourcemanager.webapp.https.address.rm2
   - yarn.resourcemanager.cluster-id
   - yarn.resourcemanager.ha.automatic-failover.zk-base-path
4. Verify that the following properties in the yarn-site.json file are set to the ResourceManager hostname you are keeping:
   - yarn.resourcemanager.hostname
   - yarn.resourcemanager.admin.address

- yarn.resourcemanager.webapp.address
- yarn.resourcemanager.resource-tracker.address
- yarn.resourcemanager.scheduler.address
- yarn.resourcemanager.webapp.https.address
- yarn.timeline-service.webapp.address
- yarn.timeline-service.webapp.https.address
- yarn.timeline-service.address
- yarn.log.server.url

5. Search the yarn-site.json file and remove any references to the ResourceManager hostname that you are removing.
6. Search the yarn-site.json file and remove any properties that might still be set for ResourceManager IDs.
   For example, rm1 and rm2.
7. Save the yarn-site.json file and set that configuration against the Ambari Server.
   /var/lib/ambari-server/resources/scripts/configs.py set [AMBARI_SERVER] [CLUSTER_NAME] yarn-site yarn-site.json
8. Using the Ambari API, delete the ResourceManager host component for the host that you are deleting.
   curl --user admin:admin -i -H "X-Requested-By: ambari" -X DELETE http://ambari.server:8080/api/v1/clusters/cluster.name/hosts/hostname/host_components/RESOURCEMANAGER
9. In **Ambari Web**, start the ZooKeeper service.
10. On a host that has the ZooKeeper client installed, use the ZooKeeper client to change znode permissions.

```
/usr/hdp/current/zookeeper-client/bin/zkCli.sh
getAcl /rmstore/ZKRMStateRoot
setAcl /rmstore/ZKRMStateRoot world:anyone:rwcda
```

11. In **Ambari Web**, restart ZooKeeper service and start YARN service.

What to do next

Review and confirm all recommended configuration changes.

Configuring HBase high availability

Ambari enables simple setup of multiple HBase Masters.

To help you achieve redundancy for high availability in a production environment, Apache HBase supports deployment of multiple HBase Masters in a cluster. If you are working in an HDP 2.2 or later environment, Apache Ambari enables simple setup of multiple HBase Masters. Hortonworks recommends that you use Ambari to configure multiple HBase Masters.

During the Apache HBase service installation and depending on your component assignment, Ambari installs and configures one HBase Master component and multiple RegionServer components. To configure high availability for the HBase service, you can run two or more HBase Master components. HBase uses ZooKeeper for coordination of the active Master in a cluster running two or more HBase Masters. This means, when active HBase Master fails, the client will be automatically routed to standby Master.

Add a new HBase master to an existing cluster

Procedure

1. Log in to the Ambari management interface as a cluster administrator.
   admin/admin
2. In **Ambari Web**, browse to **Services** > **HBase**.
3. In **Service Actions**, click **+ Add HBase Master**.
4. Choose the host on which to install the additional HBase master; then click **Confirm Add**.

Results

Ambari installs the new HBase Master and reconfigures HBase to manage multiple Master instances.

What to do next

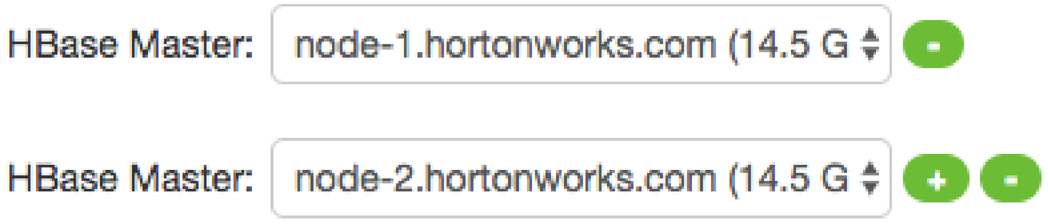
Add a standby HBase master to the cluster.

Add a standby HBase master to a new cluster

During cluster deployment, you can add a standby HBase master when you add the active HBase master.

Procedure

- Click the + sign that is displayed to the right of the HBase Master host name. of the existing



HBase Master: node-1.hortonworks.com (14.5 G

HBase Master: node-2.hortonworks.com (14.5 G

The first host you add will be the active node. The second will be the standby HBase Master.

Setting Up Multiple HBase Masters Manually

Set up your active HBase master using the cluster deployment wizard. If necessary, add additional HBase masters manually.

About this task

You must configure the first node (node-1) on your cluster by following the instructions in the cluster deployment topic.

Configure passwordless ssh access for HBase

About this task

The first node on the cluster (node-1) must be able to log in to other nodes on the cluster and then back to itself in order to start the daemons. You can accomplish this by using the same user name on all hosts and by using passwordless Secure Socket Shell (SSH) login.

Procedure

1. On node-1, stop HBase service.
2. On node-1, log in as an HBase user and generate an SSH key pair.
   $ ssh-keygen -t rsa
   The system prints the location of the key pair to standard output. The default name of the public key is id_rsa.pub.
3. Create a directory to hold the shared keys on the other nodes.
   a) On node-2, log in as an HBase user and create an .ssh/ directory in your home directory.
   b) On node-3, log in as an HBase user and create an .ssh/ directory in your home directory.
4. Use Secure Copy (scp) or any other standard secure means to copy the public key from node-1 to the other two nodes. On each node in the cluster, create a new file called .ssh/authorized_keys (if it does not already exist) and append the contents of the id_rsa.pub file to it.
   $ cat id_rsa.pub >> ~/.ssh/authorized_keys

Ensure that you do not overwrite your existing .ssh/authorized_keys files by concatenating the new key onto the existing file using the >> operator rather than the > operator.

5. Use Secure Shell (SSH) from node-1 to either of the other nodes using the same user name.

   You should not be prompted for password.
6. On node-2, repeat Step 5, because it runs as a backup Master.

Prepare node-1 for Hbase

Prevent the RegionServer on node-1 from starting.

About this task

Because node-1 should run your active Master and ZooKeeper processes, you must stop the RegionServer from starting on node-1.

Procedure

1. Edit conf/regionservers by removing the line that contains localhost and adding lines with the host name or IP addresseses for node-2 and node-3.

   If you want to run a RegionServer on node-1, you should refer to it by the hostname the other servers would use to communicate with it.

   For example, for node-1, it is called as node-1.test.com.
2. Configure HBase to use node-2 as a backup Master by creating a new file in conf/ called backup-Masters, and adding a new line to it with the host name for node-2.
   For example, node-2.test.com.
3. Configure ZooKeeper on node-1 by editing conf/hbase-site.xml and adding the following properties:

   ```
   <property>
   <name>hbase.zookeeper.quorum</name>
   <value>node-1.test.com,node-2.test.com,node-3.test.com</value>
   </property>
   <property>
   <name>hbase.zookeeper.property.dataDir</name>
   <value>/usr/local/zookeeper</value>
   </property>
   ```

   This configuration directs HBase to start and manage a ZooKeeper instance on each node of the cluster. You can learn more about configuring ZooKeeper at the Apache Zookeeper project site.
4. Change every reference in your configuration to node-1 as localhost to point to the host name that the other nodes use to refer to node-1.
   In this example, node-1.test.com.

Prepare node-2 and node-3 for Hbase

About this task

Before preparing node-2 and node-3, each node of your cluster must have the same configuration information. Node-2 runs as a backup Master server and a ZooKeeper instance.

Procedure

1. Download and unpack HBase on node-2 and node-3.
2. Copy the configuration files from node-1 to node-2 and node-3.
3. Copy the contents of the conf/ directory to the conf/ directory on node-2 and node-3.

Start and test your HBase cluster

Procedure

1. Use the jps command to ensure that HBase is not running.
2. Kill HMaster, HRegionServer, and HQuorumPeer processes, if they are running.
3. Start the cluster by running the start-hbase.sh command on node-1.
   Your output is similar to this:

```
$ bin/start-hbase.sh
node-3.test.com: starting zookeeper, logging to /home/hbuser/hbase-0.98.3-
hadoop2/bin/../logs/hbase-hbuser-zookeeper-node-3.test.com.out
node-1.example.com: starting zookeeper, logging to /home/
hbuser/hbase-0.98.3-hadoop2/bin/../logs/hbase-hbuser-zookeeper-
node-1.test.com.out
node-2.example.com: starting zookeeper, logging to /home/
hbuser/hbase-0.98.3-hadoop2/bin/../logs/hbase-hbuser-zookeeper-
node-2.test.com.out
starting master, logging to /home/hbuser/hbase-0.98.3-hadoop2/bin/../logs/
hbase-hbuser-master-node-1.test.com.out
node-3.test.com: starting regionserver, logging to /home/hbuser/
hbase-0.98.3-hadoop2/bin/../logs/hbase-hbuser-regionserver-
node-3.test.com.out
node-2.test.com: starting regionserver, logging to /home/hbuser/
hbase-0.98.3-hadoop2/bin/../logs/hbase-hbuser-regionserver-
node-2.test.com.out
node-2.test.com: starting master, logging to /home/hbuser/hbase-0.98.3-
hadoop2/bin/../logs/hbase-hbuser-master-node2.test.com.out
```

   ZooKeeper starts first, followed by the Master, then the RegionServers, and finally the backup Masters.
4. Run the jps command on each node to verify that the correct processes are running on each server.
   Example1. node-1 jps Output

```
$ jps
20355 Jps
20071 HQuorumPeer
20137 HMaster
```

   Example 2. node-2 jps Output

```
$ jps
15930 HRegionServer
16194 Jps
15838 HQuorumPeer
16010 HMaster
```

   Example 3. node-3 jps Output

```
$ jps
13901 Jps
13639 HQuorumPeer
13737 HRegionServer
```

   ZooKeeper Process Name

   You might see additional Java processes running on your servers as well, if they are used for any other purposes.

   The HQuorumPeer process is a ZooKeeper instance which is controlled and started by HBase. If you use ZooKeeper this way, it is limited to one instance per cluster node and is appropriate for testing only. If ZooKeeper is run outside of HBase, the process is called QuorumPeer. For more about ZooKeeper configuration, including using an external ZooKeeper instance with HBase, see the zookeeper project site.

5. Browse to the Web UI and test your new connections.

   You should be able to connect to the UI for the Master http://node-1.test.com:16010/ or the standby master at http://node-2.test.com:16010/. If you can connect through localhost but not from another host, check your firewall rules. You can see the web UI for each of the RegionServers at port 16030 of their IP addresses, or by clicking their links in the web UI for the Master.

   Web UI Port Changes

   In HBase newer than 0.98.x, the HTTP ports used by the HBase Web UI changed from 60010 for the Master and 60030 for each RegionServer to 16010 for the Master and 16030 for the RegionServer.

Configuring Hive high availability

The Apache Hive service has multiple, associated components. The active Hive components are Hive Metastore and HiveServer2. You can configure high availability for the Hive service in HDP 2.2 or later by running two or more of each of those components. The relational database that backs the Hive Metastore itself should also be made highly available using best practices defined for the database system in use and should be done after consultation with your in-house DBA.

Add a Hive metastore component

Before you begin

If you have ACID enabled in Hive, ensure that the Run Compactor setting is enabled (set to True) on only one Hive metastore host.

Procedure

1. In **Ambari Web**, browse to **Services** > **Hive**.
2. In **Service Actions**, click the **+ Add Hive Metastore** option.
3. Choose the host on which to install the additional Hive Metastore; then click **Confirm Add**.

Results

Ambari installs the component and reconfigures Hive to handle multiple Hive Metastore instances.

What to do next

Review and confirm all recommended configuration changes.

Add a HiveServer2 component

Procedure

1. In **Ambari Web**, browse to the host on which you want to install another HiveServer2 component.
2. On the **Host** page, click **+Add**.
3. Click **HiveServer2** from the list.

Results

Ambari installs the additional HiveServer2 component.

What to do next

Review and confirm all recommended configuration changes.

Add a WebHCat server

Procedure

1. In **Ambari Web**, browse to the host on which you want to install another WebHCat Server.
2. On the **Host** page, click **+Add**.
3. Click **WebHCat** from the list.

Results

Ambari installs the new server and reconfigures Hive to manage multiple Metastore instances.

What to do next

Review and confirm all recommended configuration changes.

Configuring Storm high availability

Use Ambari to configure HA for Storm, by adding a Nimbus component.

In HDP 2.3 or later, you can use Ambari to configure high availability for the Apache Storm Nimbus server by adding aan additional, standby Nimbus component.

Add a Nimbus Component

Procedure

1. In **Ambari Web**, browse to **Services** > **Storm**.
2. In **Service Actions**, click the **+ Add Nimbus** option.
3. Click the host on which to install the additional Nimbus; then click **Confirm Add**.

Results

Ambari installs the component and reconfigures Storm to handle multiple Nimbus instances.

What to do next

Review and confirm all recommended configuration changes.

Configuring Oozie high availability

To set up high availability for the Apache Oozie service in HDP 2.2 or later, you can run two or more instances of the Oozie Server component.

Add an Oozie server component

Before you begin

- The relational database that backs the Oozie Server should also be made highly available using best practices defined for the database system in use and should be done after consultation with your in-house DBA. Using the default installed Derby database instance is not supported with multiple Oozie Server instances; therefore, you must use an existing relational database. When using Apache Derby for the Oozie Server, you do not have the option to add Oozie Server components to your cluster.
- High availability for Oozie requires the use of an external virtual IP address or load balancer to direct traffic to the Oozie servers.

Procedure

1. In **Ambari Web**, browse to the host on which you want to install another Oozie Server.

2. On the **Host** page, click **+Add**.
3. Click **Oozie Server** from the list.
   Ambari installs the new Oozie Server.
4. Configure your external load balancer.
5. Update the Oozie configuration.
   a) Browse to **Services** > **Oozie** > **Configs**.
   b) In oozie-site, add the following property values:

oozie.zookeeper.connection.string	List of ZooKeeper hosts with ports: for example, c6401.ambari.apache.org:2181, c6402.ambari.apache.org:2181, c6403.ambari.apache.org:2181
oozie.services.ext	org.apache.oozie.service.ZKLocksService, org.apache.oozie.service.ZKXLogStreamingService, org.apache.oozie.service.ZKJobsConcurrencyService
oozie.base.url	http://[CLOADBALANCER_HOSTNAME]:11000/oozie

   c) In oozie-env, uncomment the oozie_base_url property and change its value to point to the load balancer.
   export oozie_base_url="http://<loadbalance.hostname>:11000/oozie"
6. Restart Oozie.
7. Update the HDFS configuration properties for the Oozie proxy user.
   a) Browse to **Services** > **HDFS** > **Configs**.
   b) In core-site, update the hadoop.proxyuser.oozie.hosts property to include the newly added Oozie Server host.
   Use commas to separate multiple host names.
8. Restart services.

What to do next

Review and confirm all recommended configuration changes.

Configuring Atlas high availability

Before you begin

In Ambari 2.4.0.0, adding or removing Atlas Metadata Servers requires manually editing the atlas.rest.address property.

Procedure

1. In **Ambari Web**, click **Hosts**, then select the host on which to install the standby Atlas Metadata Server.
2. On the **Summary** page of the new Atlas Metadata Server host, click **Add** > **Atlas Metadata Server**.
3. Complete the wizard steps to add the new Atlas Metadata Server.
   Ambari adds the new Atlas Metadata Server in a Stopped state.
4. On **Services**, browse to **Atlas** > **Configs** > **Advanced**.
5. In **Advanced application-properties**, append the atlas.rest.address property with a comma and the value for the new Atlas Metadata Server.
   atlas.rest.address property,http(s):[HOST_NAME]:[PORT_NUMBER]

The default protocol is http. If the atlas.enableTLS property is set to true, use https. The default HTTP port is 21000 and the default HTTPS port is 21443. These values can be overridden using the atlas.server.http.port and atlas.server.https.port properties, respectively.

6. Stop all running Atlas Metadata Servers.

   Important: You must use the **Stop** command to stop the Atlas Metadata Servers. Do not use a **Restart** command. This attempts to first stop the newly added Atlas Server, which at this point does not contain any configurations in /etc/atlas/conf.

7. In **Ambari Web**, browse to **Services** > **Atlas** > **Service Actions**, then, click **Start**.
   Ambari automatically configures the following Atlas properties in the /etc/atlas/conf/atlas-application.properties file:

   - atlas.server.ids
   - atlas.server.address.$id
   - atlas.server.ha.enabled

8. To refresh the configuration files, restart the following services that contain Atlas hooks:

   - Hive
   - Storm
   - Falcon
   - Sqoop
   - Oozie

9. In **Services**, browse to **Actions** > **Restart All Required**.

   When you update the Atlas configuration settings in Ambari, Ambari marks the services that require restart.

10. Browse to **Oozie** > **Service Actions**, then click **Restart All**.

    Apache Oozie requires a restart after an Atlas configuration update, but may not be included in the services marked as requiring restart in Ambari.

What to do next

Review and confirm all recommended configuration changes.

Related Information

Review and confirm configuration changes

Restart all required services

Enabling Ranger admin high availability

You can configure Ranger Admin high availability (HA) with or without SSL on an Ambari-managed cluster. Please note that the configuration settings used in this section are sample values. You should adjust these settings to reflect your environment (folder locations, passwords, file names, and so on).

Related Information

Configuring Apache Ranger High Availability

Review and confirm configuration changes

Restart all required services