Cloudera Manager 7.4.0

Configuring Clusters

Date published: 2020-11-30
Date modified: 2021-03-24



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2024. All rights reserved.

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

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

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

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

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

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

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

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

Contents

Modifying Configuration Properties Using Cloudera Manager..... 5
Changing the Configuration of a Service or Role Instance.....5
Searching for Properties..... 8
Validation of Configuration Properties..... 8
Overriding Configuration Properties..... 8
Viewing and Editing Overridden Configuration Properties.....8
Resetting Configuration Properties to the Default Value..... 9
Viewing and Editing Host Overrides.....9
Restarting Services and Instances after Configuration Changes..... 10
Suppressing Configuration and Parameter Validation Warnings..... 11
Suppressing a Configuration Validation in Cloudera Manager..... 11
Managing Suppressed Validations..... 11
Suppressing Configuration Validations Before They Trigger Warnings..... 12
Viewing a List of All Suppressed Validations..... 12

Cluster-Wide Configuration..... 12

Custom Configuration.....13

Setting an Advanced Configuration Snippet for a Cloudera Runtime Service..... 15

Setting an Advanced Configuration Snippet for a Cluster.....17

Stale Configurations..... 19

Client Configuration Files..... 21
How Client Configurations are Deployed.....21
Downloading Client Configuration Files..... 22
Manually Redeploying Client Configuration Files..... 22

Viewing and Reverting Configuration Changes.....22
Viewing and reverting configuration changes for a service, role, or host.....23
Viewing and reverting configuration changes for a cluster.....23

Autoconfiguration..... 23
Autoconfiguration..... 24
Configuration Scope..... 24
Data Directories..... 24
Memory..... 25

General Rules....29
Role-Host Placement....30

Using the Cloudera Manager API....31
Using the Cloudera Manager API to backup and restore clusters....31
Backing up the Cloudera Manager configuration....31
Restoring the Cloudera Manager configuration....32
Using the Cloudera Manager API to Manage and Configure Clusters....33
Using the Cloudera Manager API for Cluster Automation....33
Using the Cloudera Manager API to Obtain Configuration Files....35
Using the Cloudera Manager API to Set Advanced Configuration Snippets (Safety Valves)....37
Using Tags in Cloudera Manager....38
Initiating HDFS failover using the Cloudera Manager API....41
Creating a Runtime Cluster Using a Cloudera Manager Template....42
Exporting the Cluster Configuration....43
Preparing a New Cluster....44
Creating the Template....44
Importing the Template to a New Cluster....49
Sample Python Code....50

Modifying Configuration Properties Using Cloudera Manager

Using Cloudera Manager to modify cluster configurations.

When a service is added to Cloudera Manager, either through the installation or upgrade wizard or with the Add Services workflow, Cloudera Manager automatically sets the configuration properties, based on the needs of the service and characteristics of the cluster in which it will run. These configuration properties include both service-wide configuration properties, as well as specific properties for each role type associated with the service, managed through role groups. A role group is a set of configuration properties for a role type, as well as a list of role instances associated with that group. Cloudera Manager automatically creates a default role group named Role Type Default Group for each role type.

Related Information

Role Groups

Changing the Configuration of a Service or Role Instance

About this task

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Procedure

1. In the left menu, click Clusters *service name*.
2. Click the Configuration tab.
3. Locate the property you want to edit. You can type all or part of the property name in the search box, or use the Filters panel on the left side of the screen. Click the Filters toggle to show or hide the Filters panel. Click the Clear link to clear any selected filters.
   - Scope

     The Scope section in the Filters panel organizes the configuration properties by role types; first those that are Service-Wide, followed by various role types within the service. When you select one of these roles, a set of properties whose values are managed by the default role group for the role display. Any additional role groups that apply to the property also appear in this panel and you can modify values for each role group just as you can the default role group.
   - Category

     The Category section in the Filters panel allows you to limit the displayed properties by category.
   - Status

     The Status section in the Filters panel limits the displayed properties by their status. Possible statuses include:

     - Error
     - Warning
     - Edited
     - Non-default
     - Has Overrides

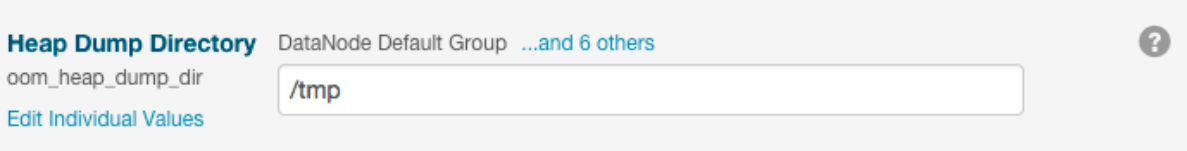
4. Edit the property value.
   - To facilitate entering some types of values, you can specify not only the value, but also the units that apply to the value. for example, to enter a setting that specifies bytes per second, you can choose to enter the value in bytes (B), KiBs, MiBs, or GiBs—selected from a drop-down menu that appears when you edit the value.
   - If the property allows a list of values, click the ✚ icon to the right of the edit field to add an additional field. An example of this is the HDFS DataNode Data Directory property, which can have a comma-delimited list of

directories as its value. To remove an item from such a list, click the − icon to the right of the field you want to remove.

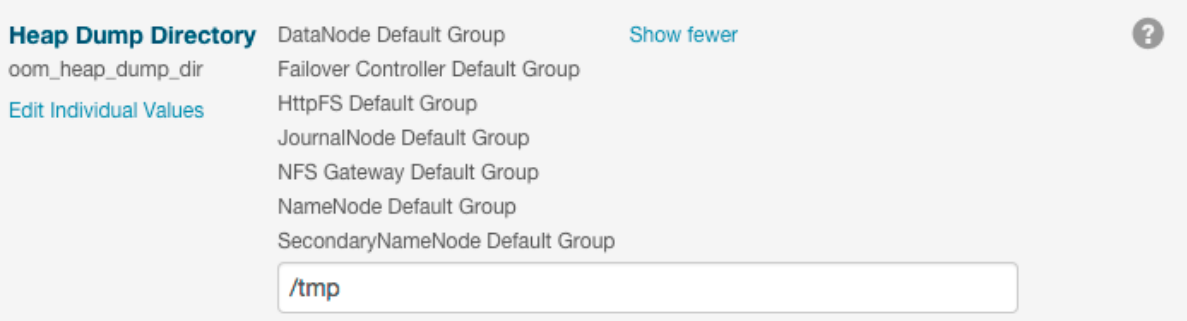
Many configuration properties have different values that are configured by multiple role groups.

To edit configuration values for multiple role groups:

a) Go to the property, For example, the configuration panel for the Heap Dump Directory property displays the DataNode Default Group (a role group), and a link that says ... and 6 others.

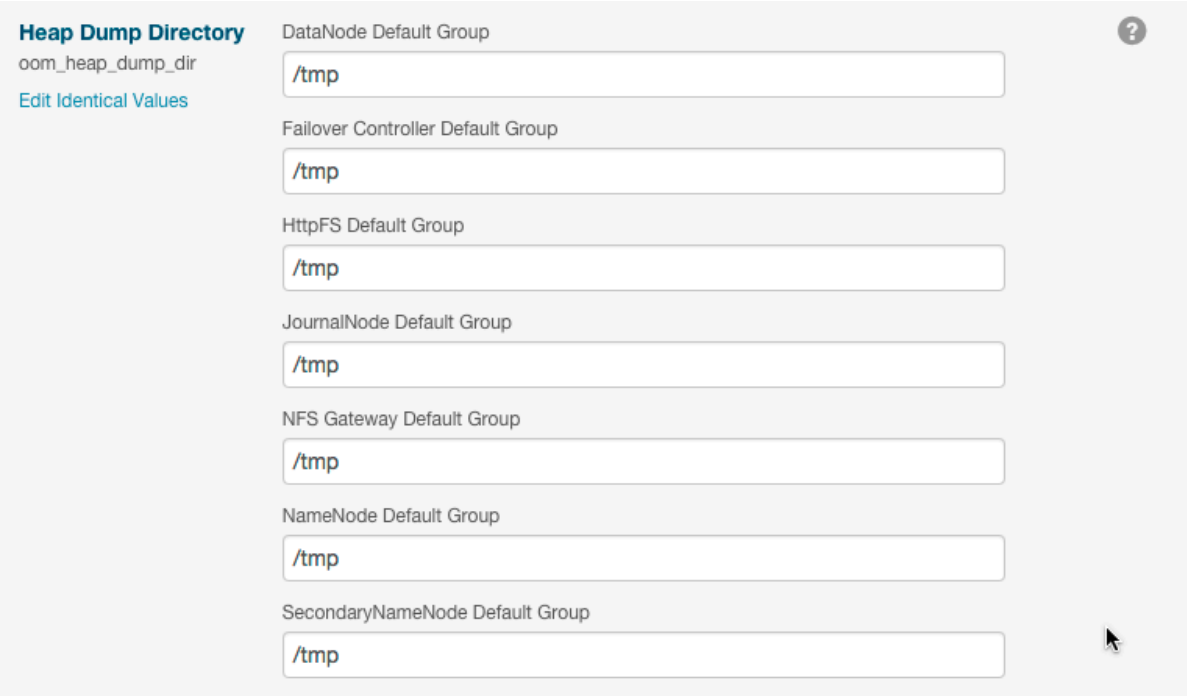


b) Click the ... and 6 others link to display all of the role groups:



c) Click the Show fewer link to collapse the list of role groups.

If you edit the single value for this property, Cloudera Manager applies the value to all role groups. To edit the values for one or more of these role groups individually, click Edit Individual Values. Individual fields display where you can edit the values for each role group. For example:



5. The Reasons for Change field is automatically populated with information about configuration changes you have made. You can edit or add any additional information. This information is saved with the configuration history, which you can view by clicking the History and Rollback link in the top portion of the screen.

6. Click Save Changes (or CNTRL + S) to commit the changes.

   This changes the setting for the role group, and applies to all role instances associated with that role group. Depending on the change you made, you may need to restart the service or roles associated with the configuration you just changed. Or, you may need to redeploy your client configuration for the service. You should see a message to that effect at the top of the Configuration page, and services will display an outdated configuration (Restart Needed), (Refresh Needed), or outdated client configuration indicator. Click the indicator to display the **Stale Configurations** page.

What to do next

To view or revert your configuration changes, click the History and Rollback link in the top portion of the screen.

Related Information

Role Groups

Stale Configurations

Searching for Properties

You can use the Search box to search for properties by name or label. The search also returns properties whose description matches your search term.

Validation of Configuration Properties

Cloudera Manager validates the values you specify for configuration properties. If you specify a value that is outside the recommended range of values or is invalid, Cloudera Manager displays a warning at the top of the Configuration tab and in the text box after you click Save Changes. The warning is yellow if the value is outside the recommended range of values and red if the value is invalid.

Overriding Configuration Properties

About this task

For role types that allow multiple instances, each role instance inherits its configuration properties from its associated role group. While role groups provide a convenient way to provide alternate configuration properties for selected groups of role instances, there may be situations where you want to make a one-off configuration change—for example when a host has malfunctioned and you want to temporarily reconfigure it. In this case, you can override configuration properties for a specific role instance:

Procedure

1. Go to the page of the service with the role you want to change. Click Clusters in the left menu and select the service, or click the Status tab on the Cloudera Manager Home page and select the service.
2. Click the Instances tab.
3. Click the role instance you want to change.
4. Click the Configuration tab.
5. Change the configuration values as appropriate.
6. Save your changes.

What to do next

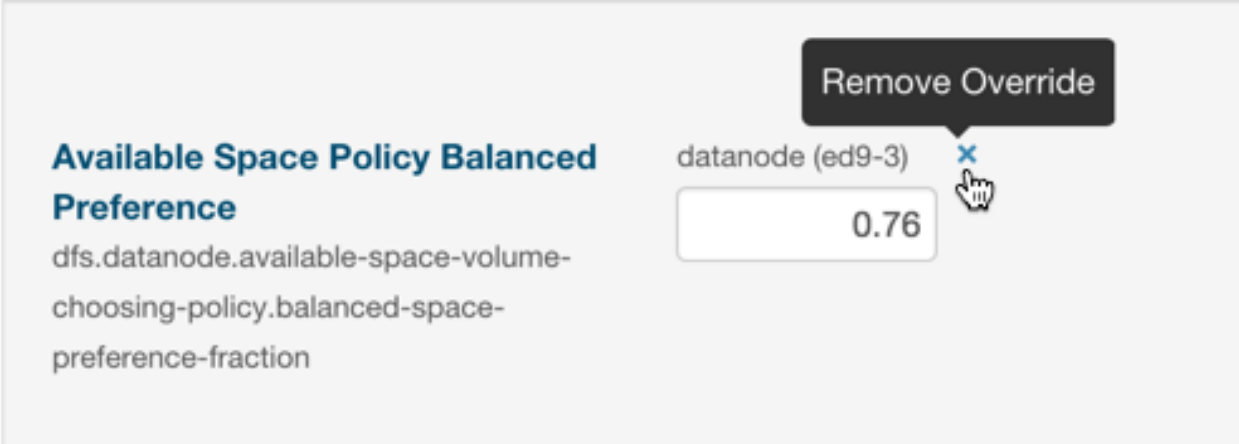
You will most likely need to restart your service or role to have your configuration changes take effect.

Related Information

Stale Configurations

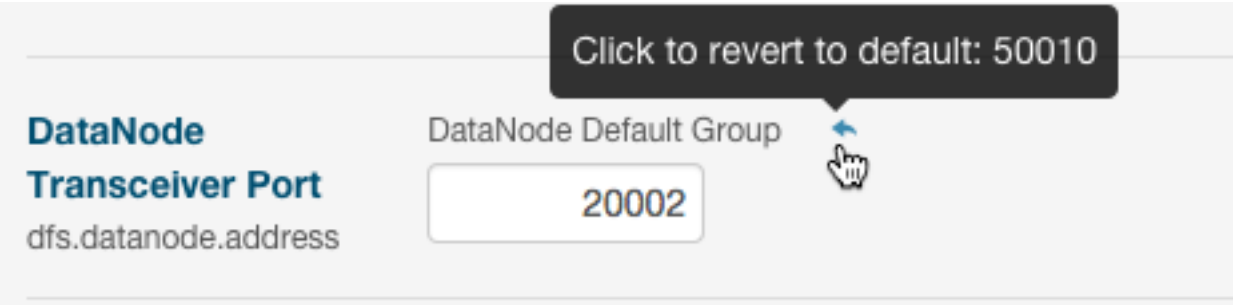
Viewing and Editing Overridden Configuration Properties

To see a list of all role instances that have an override value for a particular configuration setting, click the Configuration tab on the page for the service and select Status Has overrides . A list of configuration properties where values have been overridden displays. The panel for each configuration property displays the values for each role group or instance. You can edit the value of this property for this instance, or, you can click the icon next to an instance name to remove the overridden value.



Resetting Configuration Properties to the Default Value

To reset a property back to its default value, click the icon. The default value is inserted and the icon turns into an Undo icon ( ). Explicitly setting a configuration to the same value as its default (inherited value) has the same effect as using the icon.



There is no mechanism for resetting to an autoconfigured value. However, you can use the configuration history and rollback feature to revert any configuration changes.

Related Information

Autoconfiguration
Viewing and Reverting Configuration Changes

Viewing and Editing Host Overrides

You can override the properties of individual hosts in your cluster.

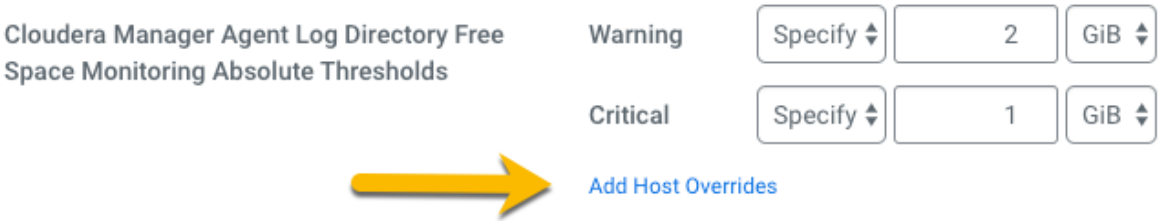
Before you begin

Minimum Required Role: Cluster Administrator (also provided by Full Administrator) This feature is not available when using Cloudera Manager to manage Data Hub clusters.

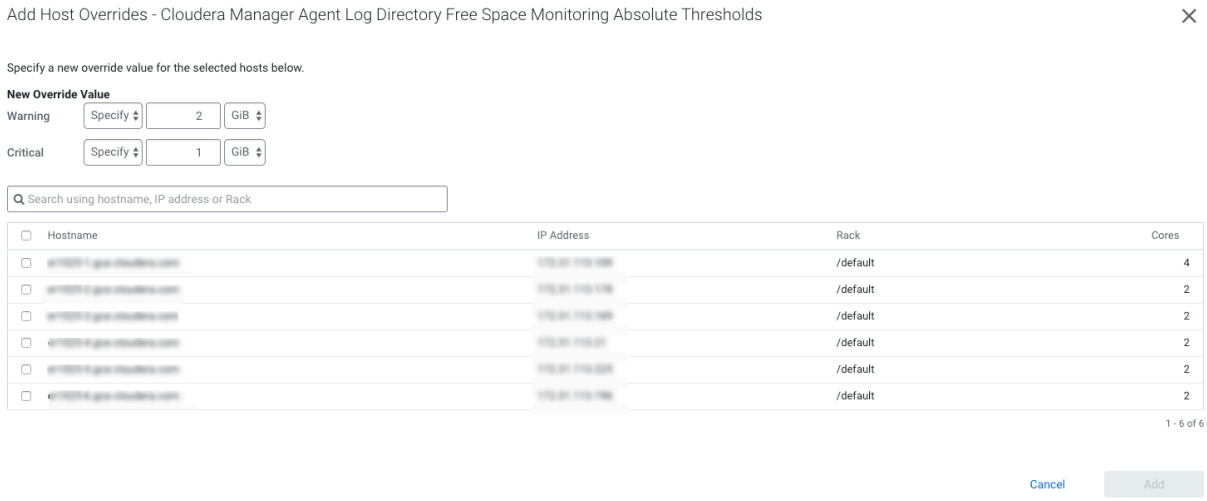
Procedure

1. In the left menu, click HostsConfiguration.

2. Use the Filters or Search box to locate the property that you want to override.
3. Click the Add Host Overrides link.



The Add Host Overrides dialog box displays.



4. Select one or more hosts to override this property.
5. Enter the new value(s) in the New Override Value section.
6. Click Add.
   The Hosts Configuration page displays again and now shows the overridden values you just entered.
7. If necessary, you can further edit the overrides:
   - To remove the override, click the X next to the Remove Override link.
   - To apply the same value to all hosts, click Edit Identical Values. Click Edit Individual Values to apply different values to selected hosts.
8. Click Save Changes.
9. If the property indicates Requires Agent Restart, restart the agent on the affected hosts.

Restarting Services and Instances after Configuration Changes

If you change the configuration properties after you start a service or instance, you may need to restart the service or instance to have the configuration properties become active.

About this task

Minimum Required Role: Operator (also provided by Configurator, Cluster Administrator, Limited Cluster Administrator , Full Administrator)

If you change configuration properties at the service level that affect a particular role only (such as all DataNodes but not the NameNodes), you can restart only that role; you do not need to restart the entire service. If you changed the configuration for a particular role instance (such as one of four DataNodes), you may need to restart only that instance.

Procedure

1. Follow the instructions in the topics *Restarting a Service* or *Starting, Stopping, and Restarting Role Instances*.
2. If you see a Finished status, the service or role instances have restarted.
3. Go to the Home Status tab.

Results

The service should show a Status of Started for all instances and a health status of Good.

Related Information

Restarting a Cloudera Runtime Service
Starting, Stopping, and Restarting Role Instances
Stale Configurations

Suppressing Configuration and Parameter Validation Warnings

Suppressing configuration validation warnings. d

You can suppress the warnings that Cloudera Manager issues when a configuration value is outside the recommended range or is invalid. If a warning does not apply to your deployment, you might want to suppress it. Suppressed validation warnings are still retained by Cloudera Manager, and you can unsuppress the warnings at any time. You can suppress each warning when you view it, or you can configure suppression for a specific validation before warnings occur.

Suppressing a Configuration Validation in Cloudera Manager

When viewing the configuration issues, you can suppress each warning. A dialog box opens where you can enter a comment about the suppression.

About this task

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Procedure

To suppress warnings from the All Configuration Issues screen:

1. Click the Cloudera Manager logo to return to the Home screen.
2. Click Configuration Configuration Issues .
3. Locate the validation message in the list and click the Suppress... link.

   A dialog box opens where you can enter a comment about the suppression.
4. Click Confirm.

Managing Suppressed Validations

About this task

On pages where you have suppressed validations, you see a link that says Show # Suppressed Warning(s). On this screen, you can:

- Click the Show # Suppressed Warning(s) link to show the warnings.

  Each suppressed warning displays an icon: .
- Click the Unsuppress... link to unsuppress the configuration validation.
- Click the Hide Suppressed Warnings link to re-hide the suppressed warnings.

Suppressing Configuration Validations Before They Trigger Warnings

Procedure

1. Go to the service or host with the configuration validation warnings you want to suppress.
2. Click Configuration .
3. In the filters on the left, select Category Suppressions .

   A list of suppression properties displays. The names of the properties begin with Suppress Parameter Validation or Suppress Configuration Validator. You can also use the Search function to limit the number of properties that display.
4. Select a suppression property to suppress the validation warning.
5. Enter a Reason for change, and then click Save Changes to commit the changes.

Viewing a List of All Suppressed Validations

About this task

Do one of the following:

- From the Home page or the Status page of a cluster, select Configuration Suppressed Health and Configuration Issues .
- From the Status page of a service, select Configuration Category Suppressions and select Status Non-default .
- From the left menu, select HostsHosts Configuration, then Category Suppressions and select Status Non-default.

Cluster-Wide Configuration

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

To make configuration changes that apply to an entire cluster, do one of the following to open the configuration page:

- All Clusters

  1. From the Cloudera Manager Home page, click Configuration and then select one of the following classes of properties:

     - Advanced Configuration Snippets
     - Database Settings
     - Disk Space Thresholds
     - Local Data Directories and Files
     - Log Directories
     - Navigator Settings
     - Non-default Values - properties whose value differs from the default value
     - Non-uniform Values - properties whose values are not uniform across the cluster or clusters
     - Ports
     - Service Dependencies

  You can also select Configuration Issues to view a list of configuration issues for all clusters.
- Specific Cluster

  1. From the Home page, click a cluster name, or click Clusters*Cluster name* from the left menu.
  2. Select Configuration and then select one of the classes of properties listed above.

You can also apply the following filters to limit the displayed properties:

- Enter a search term in the Search box to search for properties by name or description.
- Expand the Status filter to select options that limit the displayed properties to those with errors or warnings, properties that have been edited, properties with non-default values, or properties with overrides. Select All to remove any filtering by Status.
- Expand the Scope filter to display a list of service types. Expand a service type heading to filter on Service-Wide configurations for a specific service instance or select one of the default role groups listed under each service type. Select All to remove any filtering by Scope.
- Expand the Category filter to filter using a sub-grouping of properties. Select All to remove any filtering by Category.

Custom Configuration

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Cloudera Manager exposes properties that allow you to insert custom configuration text into XML configuration, property, and text files, or into an environment. The naming convention for these properties is: *XXX* Advanced Configuration Snippet (Safety Valve) for *YYY* or *XXX YYY* Advanced Configuration Snippet (Safety Valve), where *XXX* is a service or role and *YYY* is the target.

The values you enter into a configuration snippet must conform to the syntax of the target. For an XML configuration file, the configuration snippet must contain valid XML property definitions. For a properties file, the configuration snippet must contain valid property definitions. Some files simply require a list of host addresses.

The configuration snippet mechanism is intended for use in cases where there is configuration setting that is not exposed as a configuration property in Cloudera Manager. Configuration snippets generally override normal configuration. Contact Cloudera Support if you are required to use a configuration snippet that is not explicitly documented.

Service-wide configuration snippets apply to all roles in the service; a configuration snippet for a role group applies to all instances of the role associated with that role group.

Server and client configurations have separate configuration snippets. In general after changing a server configuration snippet you must restart the server, and after changing a client configuration snippet you must redeploy the client configuration. Sometimes you can refresh instead of restart. In some cases however, you must restart a dependent server after changing a client configuration. For example, changing a MapReduce client configuration marks the dependent Hive server as stale, which must be restarted. The Admin Console displays an indicator when a server must be restarted. In addition, the All Configuration Issues tab on the **Home** page indicates the actions you must perform to resolve stale configurations.

Configuration Snippet Types and Syntax

Configuration

Set configuration properties in various configuration files; the property name indicates into which configuration file the configuration will be placed. Configuration files have the extension .xml or .conf.

For example, there are several configuration snippets for the Hive service. One Hive configuration snippet property is called the HiveServer2 Advanced Configuration Snippet for hive-site.xml; configurations you enter here are inserted verbatim into the hive-site.xml file associated with the HiveServer2 role group.

To see a list of configuration snippets that apply to a specific configuration file, enter the configuration file name in the Search field in the top navigation bar. For example, searching for mapred-site.xml shows the configuration snippets that have mapred-site.xml in their name.

Some configuration snippet descriptions include the phrase for this role only. These configurations are stored in memory, and only inserted to the configuration when running an application from

Cloudera Manager. Otherwise, the configuration changes are added to the configuration file on disk, and are used when running the application both from Cloudera Manager and from the command line.

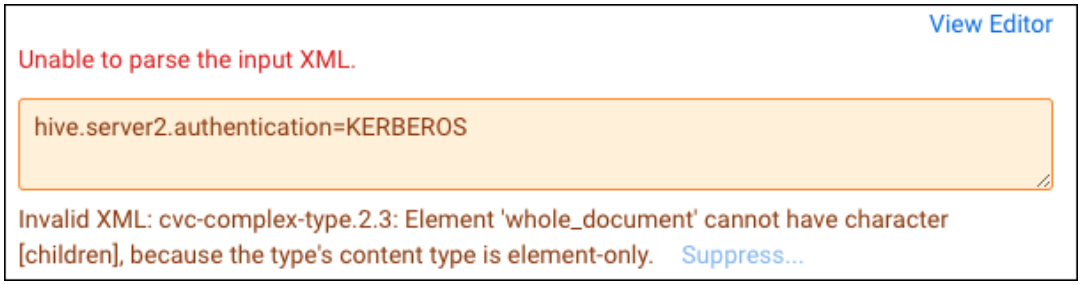
Syntax:

```
<property>
  <name>property_name</name>
  <value>property_value</value>
</property>
```

For example, to specify a MySQL connector library, put this property definition in that configuration snippet:

```
<property>
  <name>hive.aux.jars.path</name>
  <value>file:///usr/share/java/mysql-connector-java.jar</value>
</property>
```

If you do not set these properties in the correct syntax, then Cloudera Manager cannot parse the input XML and you may see an "Invalid XML" error as shown in the following image:



Environment

Specify key-value pairs for a service, role, or client that are inserted into the respective environment.

One example of using an environment configuration snippet is to add a JAR to a classpath. Place JARs in a custom location such as /opt/myjars and extend the classpath using the appropriate service environment configuration snippet. The value of a JAR property must conform to the syntax supported by its environment.

Do not place JARs inside locations such as /opt/cloudera or /usr/lib/{hadoop*,hbase*,hive*} that are managed by Cloudera because they are overwritten at upgrades.

Syntax:

```
key=value
```

For example, to add JDBC connectors to a Hive gateway classpath, add

```
AUX_CLASSPATH=/usr/share/java/mysql-connector-java.jar:\
/usr/share/java/oracle-connector-java.jar
```

or

```
AUX_CLASSPATH=/usr/share/java/*
```

to Gateway Client Advanced Configuration Snippet for hive-env.sh.

Logging

Set log4j properties in a log4j.properties file.

Syntax:

```
key1=value1
key2=value2
```

For example:

```
log4j.rootCategory=INFO, console max.log.file.size=200MB
max.log.file.backup.index=10
```

Metrics

Set properties to configure Hadoop metrics in a hadoop-metrics.properties or hadoop-metrics2.properties file.

Syntax:

```
key1=value1
key2=value2
```

For example:

```
*.sink.foo.class=org.apache.hadoop.metrics2.sink.FileSink
namenode.sink.foo.filename=/tmp/namenode-metrics.out
secondarynamenode.sink.foo.filename=/tmp/secondarynamenode-metrics.out
```

Whitelists and blacklists

Specify a list of host addresses that are allowed or disallowed from accessing a service.

Syntax:

```
host1.domain1 host2.domain2
```

Related Information

Restarting a Cloudera Runtime Service
Manually Redeploying Client Configuration Files
Stale Configurations
Cloudera Manager Admin Console
Setting the Class Path
log4j

Setting an Advanced Configuration Snippet for a Cloudera Runtime Service

How to enter service configurations using Advanced Configuration Snippets

About this task

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Procedure

1. Click Clusters in the left navigation panel and select the Cloudera Runtime service you want to configure..

2. Click the Configuration tab.
3. In the Search box, type Advanced Configuration Snippet.
4. Locate the property you want to configure. (It contains the string Advanced Configuration Snippet (Safety Valve).)

   There are two types of Advanced Configuration Snippets: XML, and Environment.

   - XML Advanced Configuration Snippet

     Use either the Snippet editor, or the XML editor:

     - Snippet editor

       Click ✚ to add a property. Enter the property name, value, and optional description. To indicate that the property value cannot be overridden by another , select the Final checkbox.

     - XML Editor

       Click the View as XML link to view and edit the XML directly. Enter the property name, value, and optional description as XML elements. To indicate that the property value cannot be overridden, specify <final>true</final>. For example:

       ```
       <property>
         <name>foo</name>
         <value>bar</value>
         <description>Example</description>
       ```

```
</property>
```

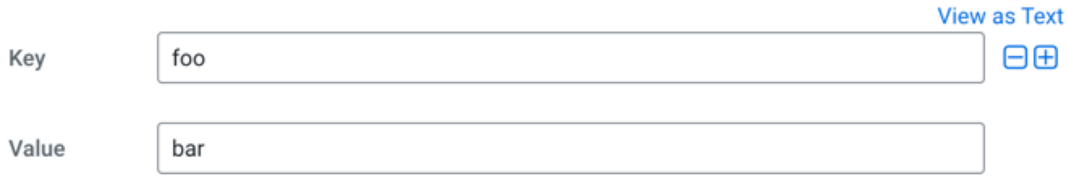
To switch between the editor and text field, click the View Editor and View as XML links at the top right of the snippet row.

- Environment Advanced Configuration Snippet

  Use either the Snippet editor, or the Text editor:

  - Snippet editor

    Click ✚ to add a property and enter the key name and value.

  - Text Editor

    Click the Edit Text link to view and edit the text. Enter the property name and value as name/value pairs. For example:

    To switch between the editor and text field, click the View Editor and View Text links at the top right of the snippet row.

5. Enter or edit the Reason for change field.
6. Click Save Changes to commit the changes.
7. Restart the service or role or redeploy client configurations as indicated.

Setting an Advanced Configuration Snippet for a Cluster

How to enter cluster configurations using Advanced Configuration Snippets

About this task

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Procedure

1. To configure a specific cluster, select a cluster from the HomeStatus page or Clusters*Cluster name* in the left menu. To configure all clusters, start on the Cloudera Manager Home page.
2. Select ConfigurationAdvanced Configuration Snippets.

3. Locate the property you want to configure. (It contains the string Advanced Configuration Snippet (Safety Valve).)

   There are two types of Advanced Configuration Snippets: XML, and Environment.

   - XML Advanced Configuration Snippet

     Use either the Snippet editor, or the XML editor:

     - Snippet editor

       Click ✚ to add a property. Enter the property name, value, and optional description. To indicate that the property value cannot be overridden by another , select the Final checkbox.

     - XML Editor

       Click the View as XML link to view and edit the XML directly. Enter the property name, value, and optional description as XML elements. To indicate that the property value cannot be overridden, specify <final>true</final>. For example:

       ```
       <property>
         <name>foo</name>
         <value>bar</value>
         <description>Example</description>
       ```

```
</property>
```

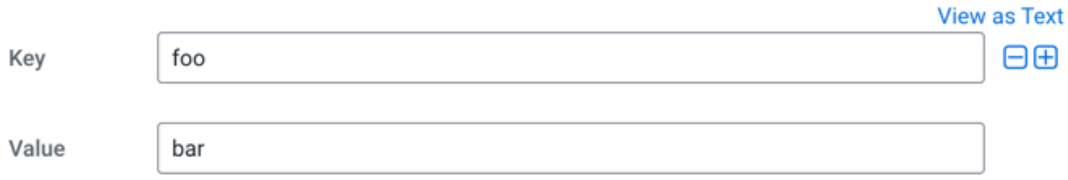
To switch between the editor and text field, click the View Editor and View as XML links at the top right of the snippet row.

- Environment Advanced Configuration Snippet

  Use either the Snippet editor, or the Text editor:

  - Snippet editor

    Click ✚ to add a property and enter the key name and value.

  - Text Editor

    Click the Edit Text link to view and edit the text. Enter the property name and value as name/value pairs. For example:

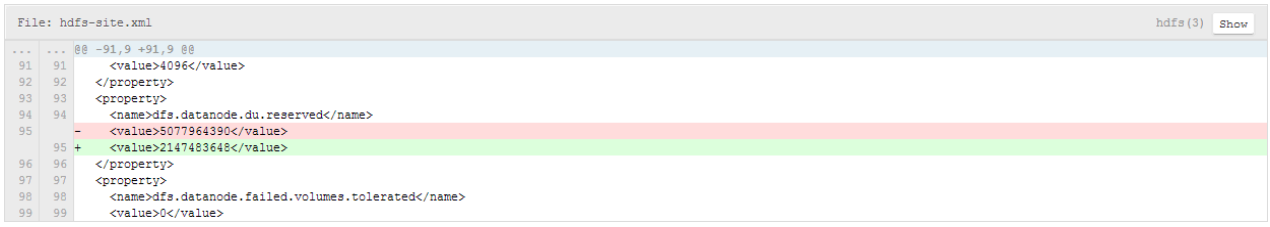
    To switch between the editor and text field, click the View Editor and View Text links at the top right of the snippet row.

4. Enter or edit the Reason for change field.
5. Click Save Changes to commit the changes.
6. Restart the service or role or redeploy client configurations as indicated.

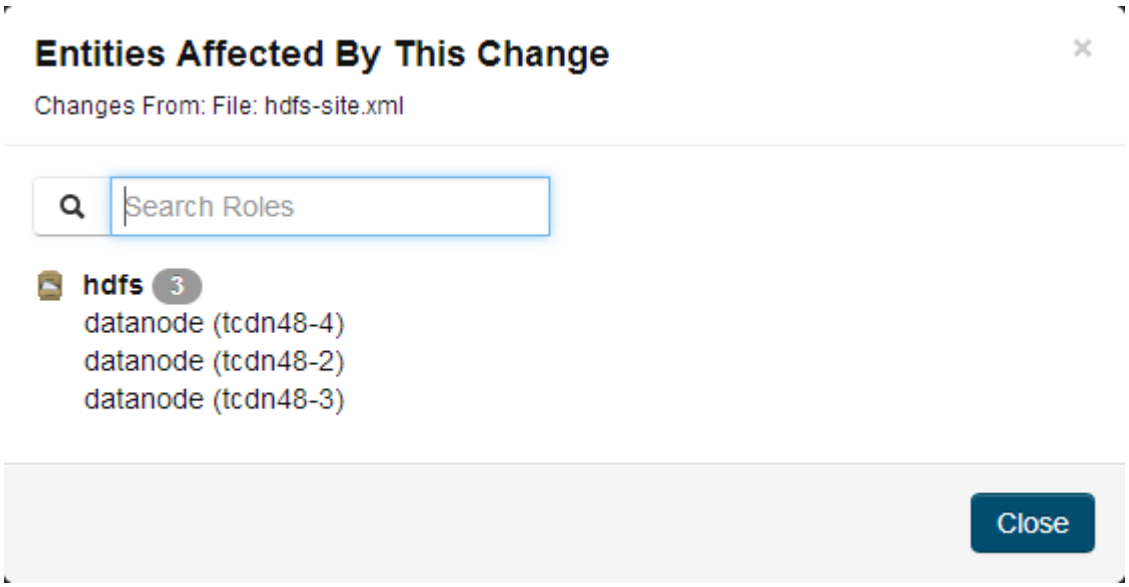
Stale Configurations

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

The Stale Configurations page provides differential views of changes made in a cluster. For any configuration change, the page contains entries of all affected attributes. For example, the following File entry shows the change to the file hdfs-site.xml when you update the property controlling how much disk space is reserved for non-HDFS use on each DataNode:



To display the entities affected by a change, click the Show button at the right of the entry. The following dialog box shows that three DataNodes were affected by the disk space change:



Viewing Stale Configurations

To view stale configurations, click the , , or indicator next to a service on the **Cloudera Manager Admin Console Home** page or on a service status page.

Attribute Categories

The categories of attributes include:

- Environment - represents environment variables set for the role. For example, the following entry shows the change to the environment that occurs when you update the heap memory configuration of the SecondaryNameNode.



- Files - represents configuration files used by the role.
- Process User & Group - represents the user and group for the role. Every role type has a configuration to specify the user/group for the process. If you change a value for a user or group on any service's configuration page it will appear in the Stale Configurations page.
- System Resources - represents system resources allocated for the role, including ports, directories, and cgroup limits. For example, a change to the port of role instance will appear in the System Resources category.
- Client Configs Metadata - represents client configurations.

Filtering Stale Configurations

You filter the entries on the Stale Configurations page by selecting from one of the drop-down lists:

- Attribute - you can filter by an attribute category such as All Files or by a specific file such as topology.map or yarn-site.xml.
- Service
- Role

After you make a selection, both the page and the drop-down show only entries that match that selection.

To reset the view, click Remove Filter or select All *XXX*, where *XXX* is Files, Services, or Roles, from the drop-down. For example, to see all the files, select All Files.

Stale Configuration Actions

The Stale Configurations page displays action buttons. The action depends on what is required to bring the entire cluster's configuration up to date. If you go to the page by clicking a (Refresh Needed) indicator, the action button will say Restart Stale Services if one of the roles listed on the page need to be restarted.

- Refresh Stale Services - Refreshes stale services.
- Restart Stale Services - Restarts stale services.
- Deploy Client Configuration - Runs the cluster deploy client configurations action.

Related Information

Cloudera Manager Admin Console
Client Configuration Files

Client Configuration Files

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

To allow clients to use the HBase, HDFS, Hive, MapReduce, and YARN services, Cloudera Manager creates zip archives of the configuration files containing the service properties. The zip archive is referred to as a *client configuration file*. Each archive contains the set of configuration files needed to access the service: for example, the MapReduce client configuration file contains copies of core-site.xml, hadoop-env.sh, hdfs-site.xml, log4j.properties, and mapred-site.xml.

Client configuration files are generated automatically by Cloudera Manager based on the services and roles you have installed and Cloudera Manager deploys these configurations automatically when you install your cluster, add a service on a host, or add a gateway role on a host. Specifically, for each host that has a service role instance installed, and for each host that is configured as a gateway role for that service, the deploy function downloads the configuration zip file, unzips it into the appropriate configuration directory, and uses the Linux alternatives mechanism to set a given, configurable priority level. If you are installing on a system that happens to have pre-existing alternatives, then it is possible another alternative may have higher priority and will continue to be used. The alternatives priority of the Cloudera Manager client configuration is configurable under the Gateway scope of the Configuration tab for the appropriate service.

You can also manually distribute client configuration files to the clients of a service.

The main circumstance that may require a redeployment of the client configuration files is when you have modified a configuration. In this case you will typically see a message instructing you to redeploy your client configurations. The affected service(s) will also display a icon. Click the indicator to display the **Stale Configurations** page.

Related Information

Gateway Roles
Alternatives Command
Stale Configurations

How Client Configurations are Deployed

Client configuration files are deployed on any host that is a client for a service—that is, that has a role for the service on that host. This includes roles such as DataNodes, TaskTrackers, RegionServers and so on as well as gateway roles for the service.

If roles for multiple services are running on the same host (for example, a DataNode role and a TaskTracker role on the same host) then the client configurations for both roles are deployed on that host, with the alternatives priority determining which configuration takes precedence.

For example, suppose we have six hosts running roles as follows: host H1: HDFS-NameNode; host H2: MR-JobTracker; host H3: HBase-Master; host H4: MR-TaskTracker, HDFS-DataNode, HBase-RegionServer; host H5: MR-Gateway; host H6: HBase-Gateway. Client configuration files will be deployed on these hosts as follows: host H1: hdfs-clientconfig (only); host H2: mapreduce-clientconfig, host H3: hbase-clientconfig; host H4: hdfs-clientconfig, mapreduce-clientconfig, hbase-clientconfig; host H5: mapreduce-clientconfig; host H6: hbase-clientconfig

If the HDFS NameNode and MapReduce JobTracker were on the same host, then that host would have both hdfs-clientconfig and mapreduce-clientconfig installed.

Downloading Client Configuration Files

Procedure

1. Go to the Cloudera Manager Admin Console Home page.
2. Click the Options menu (3 vertical dots) to the right of the cluster name and select View Client Configuration URLs.
3. Click a link or save the link URL and download the file using wget or curl.

Manually Redeploying Client Configuration Files

About this task

Although Cloudera Manager will deploy client configuration files automatically in many cases, if you have modified the configurations for a service, you may need to redeploy those configuration files.

If your client configurations were deployed automatically, the command described in this section will attempt to redeploy them as appropriate.



Note: If you are deploying client configurations on a host that has multiple services installed, some of the same configuration files, though with different configurations, will be installed in the conf directories for each service. Cloudera Manager uses the priority parameter in the alternatives --install command to ensure that the correct configuration directory is made active based on the combination of services on that host. The priority order is YARN > MapReduce > HDFS. The priority can be configured under the Gateway sections of the Configuration tab for the appropriate service.

Procedure

1. On the Home Status tab, click  to the right of the cluster name and select Deploy Client Configuration.
2. Click Deploy Client Configuration.

Viewing and Reverting Configuration Changes

View and revert configuration changes made in Cloudera Manager.

Minimum Required Role: Configurator (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Whenever you change and save a set of configuration settings for a service or role instance or a host, Cloudera Manager saves a revision of the previous settings and the name of the user who made the changes. You can then view past revisions of the configuration settings, and, if desired, roll back the settings to a previous state.

Viewing and reverting configuration changes for a service, role, or host

View and revert configuration changes for a service, role, or host.

Procedure

1. Go to the Status page for a service, role, or host
2. Click the Configuration tab. (To view configurations for all hosts, go to HostsAll HostsConfiguration.)
3. Click the History and Rollback link.
   The Configuration History opens and displays a table of configuration changes.
4. (Optional) Click Filter by Time Range and select a time range from the options on the right (30m, 1h, 2h, 6h, 12h, 1d, 7d, 30d) to limit the historical display.
5. Click the Details link.
   The Revision Details dialog box displays details of the configuration change.
6. (Optional) Click the Revert Configuration Changes button to revert the configuration change.
   Cloudera Manager may show a message indicating required restarts.

Related Information

Time Range Selector

Viewing and reverting configuration changes for a cluster

View and revert configuration changes for a cluster.

Procedure

1. Go to the Status page for the cluster.
2. Click ConfigurationConfiguration History.
   The Configuration History opens and displays a table of configuration changes.
3. (Optional) Click Filter by Time Range and select a time range from the options on the right (30m, 1h, 2h, 6h, 12h, 1d, 7d, 30d) to limit the historical display.
4. (Optional) Click the Details link in a row to view more information about the change.
   The Revision Details dialog box displays details of the configuration change.
5. (Optional) Click the Revert Configuration Changes button to revert the configuration change.
   Cloudera Manager may show a message indicating required restarts.

Autoconfiguration

Cloudera Manager provides several interactive wizards to automate common workflows:

- Installation - used to bootstrap a Cloudera Manager deployment
- Add Cluster - used when adding a new cluster
- Add Service - used when adding a new service
- Upgrade - used when upgrading to a new version of CDH
- Import MapReduce - used when migrating from MapReduce to YARN

In some of these wizards, Cloudera Manager uses a set of rules to automatically configure certain settings to best suit the characteristics of the deployment. For example, the number of hosts in the deployment drives the memory requirements for certain monitoring daemons: the more hosts, the more memory is needed. Additionally, wizards that are tasked with creating new roles will use a similar set of rules to determine an ideal host placement for those roles.

Scope

The following table shows, for each wizard, the scope of entities it affects during autoconfiguration and role-host placement.

Wizard	Autoconfiguration Scope	Role-Host Placement Scope
Installation	New cluster, Cloudera Management Service	New cluster, Cloudera Management Service
Add Cluster	New cluster	New cluster
Add Service	New service	New service
Upgrade	Cloudera Management Service	Cloudera Management Service
Import MapReduce	Existing YARN service	N/A

Certain autoconfiguration rules are unscoped, that is, they configure settings belonging to entities that aren't necessarily the entities under the wizard's scope. These exceptions are explicitly listed.

Related Information

Role Groups

Autoconfiguration

Cloudera Manager employs several different rules to drive automatic configuration, with some variation from wizard to wizard. These rules range from the simple to the complex.

Configuration Scope

One of the points of complexity in autoconfiguration is configuration scope. The configuration hierarchy as it applies to services is as follows: configurations may be modified at the service level (affecting every role in the service), role group level (affecting every role instance in the group), or role level (affecting one role instance). A configuration found in a lower level takes precedence over a configuration found in a higher level.

With the exception of the Static Service Pools, and the Import MapReduce wizard, all Cloudera Manager wizards follow a basic pattern:

1. Every role in scope is moved into its own, new, role group.
2. This role group is the receptacle for the role's "idealized" configuration. Much of this configuration is driven by properties of the role's host, which can vary from role to role.
3. Once autoconfiguration is complete, new role groups with common configurations are merged.
4. The end result is a smaller set of role groups, each with an "idealized" configuration for some subset of the roles in scope. A subset can have any number of roles; perhaps all of them, perhaps just one, and so on.

The Static Service Pools and Import MapReduce wizards configure role groups directly and do not perform any merging.

Data Directories

Several autoconfiguration rules work with data directories, and there's a common sub-rule used by all such rules to determine, out of all the mountpoints present on a host, which are appropriate for data. The subrule works as follows:

- The initial set of mountpoints for a host includes all those that are disk-backed. Network-backed mountpoints are excluded.
- Mountpoints beginning with /boot, /cdrom, /usr, /tmp, /home, or /dev are excluded.

- Mountpoints beginning with /media are excluded, unless the backing device's name contains /xvd somewhere in it.
- Mountpoints beginning with /var are excluded, unless they are /var or /var/lib.
- The largest mount point (in terms of total space, not available space) is determined.
- Other mountpoints with less than 1% total space of the largest are excluded.
- Mountpoints beginning with /var or equal to / are excluded unless they're the largest mount point.
- Remaining mountpoints are sorted lexicographically and retained for future use.

Memory

The rules used to autoconfigure memory reservations are perhaps the most complicated rules employed by Cloudera Manager. When configuring memory, Cloudera Manager must take into consideration which roles are likely to enjoy more memory, and must not over commit hosts if at all possible. To that end, it needs to consider each host as an entire unit, partitioning its available RAM into segments, one segment for each role. To make matters worse, some roles have more than one memory segment. For example, a Solr server has two memory segments: a JVM heap used for most memory allocation, and a JVM direct memory pool used for HDFS block caching. Here is the overall flow during memory autoconfiguration:

1. The set of participants includes every host under scope as well as every {role, memory segment} pair on those hosts. Some roles are under scope while others are not.
2. For each {role, segment} pair where the role is under scope, a rule is run to determine four different values for that pair:
   - Minimum memory configuration. Cloudera Manager must satisfy this minimum, possibly over-committing the host if necessary.
   - Minimum memory consumption. Like the above, but possibly scaled to account for inherent overhead. For example, JVM memory values are multiplied by 1.3 to arrive at their consumption value.
   - Ideal memory configuration. If RAM permits, Cloudera Manager will provide the pair with all of this memory.
   - Ideal memory consumption. Like the above, but scaled if necessary.
3. For each {role, segment} pair where the role is not under scope, a rule is run to determine that pair's existing memory consumption. Cloudera Manager will not configure this segment but will take it into consideration by setting the pair's "minimum" and "ideal" to the memory consumption value.
4. For each host, the following steps are taken:
   a. 20% of the host's available RAM is subtracted and reserved for the OS.
   b. sum(minimum_consumption) and sum(ideal_consumption) are calculated.
   c. An "availability ratio" is built by comparing the two sums against the host's available RAM.
      1. If RAM < sum(minimum) ratio = 0
      2. If RAM >= sum(ideal) ratio = 1
   d. If the host has more available memory than the total of the ideal memory for all roles assigned to the host, each role is assigned its ideal memory and autoconfiguration is finished.
   e. Cloudera Manager assigns all available host memory by setting each {role, segment} pair to the same consumption value, except in cases where that value is below the minimum memory or above the ideal memory for that pair. In that case, it is set to the minimum memory or the ideal memory as appropriate. This ensures that pairs with low ideal memory requirements are completely satisfied before pairs with higher ideal memory requirements.
5. The {role, segment} pair is set with the value from the previous step. In the Static Service Pools wizard, the role group is set just once (as opposed to each role).
6. Custom post-configuration rules are run.

Customization rules are applied in steps 2, 3 and 7. In step 2, there's a generic rule for most cases, as well as a series of custom rules for certain {role, segment} pairs. Likewise, there's a generic rule to calculate memory consumption in step 3 as well as some custom consumption functions for certain {role, segment} pairs.

Step 2 Generic Rule

For every {role, segment} pair where the segment defines a default value, the pair's minimum is set to the segment's minimum value (or 0 if undefined), and the ideal is set to the segment's default value.

Step 2 Custom Rules

HDFS

For the NameNode and Secondary NameNode JVM heaps, the minimum is 50 MB and the ideal is max(4 GB, sum_over_all(DataNode mountpoints' available space) / 0.000008).

MapReduce

For the JobTracker JVM heap, the minimum is 50 MB and the ideal is max(1 GB, round((1 GB * 2.3717181092 * ln(number of TaskTrackers in MapReduce service)) - 2.6019933306)). If the number of TaskTrackers <= 5, the ideal is 1 GB.

For the mapper JVM heaps, the minimum is 1 and the ideal is the number of cores, including hyperthreads, on the TaskTracker host. Memory consumption is scaled by mapred_child_java_opts_max_heap (the size of a task's heap).

For the reducer JVM heaps, the minimum is 1 and the ideal is (number of cores, including hyperthreads, on the TaskTracker host) / 2. Memory consumption is scaled by mapred_child_java_opts_max_heap (the size of a task's heap).

HBase

For the memory total allowed for HBase RegionServer JVM heap, the minimum is 50 MB and the ideal is min (31 GB ,(total RAM on region server host) * 0.64)

YARN

For the memory total allowed for containers, the minimum is 1 GB and the ideal is (total RAM on NodeManager host) * 0.64.

Hue

With the exception of the Beeswax Server (only in CDH 4), Hue roles do not have memory limits. Therefore, Cloudera Manager treats them as roles that consume a fixed amount of memory by setting their minimum and ideal consumption values, but not their configuration values. The two consumption values are set to 256 MB.

Impala

With the exception of the Impala daemon, Impala roles do not have memory limits. Therefore, Cloudera Manager treats them as roles that consume a fixed amount of memory by setting their minimum/ideal consumption values, but not their configuration values. The two consumption values are set to 150 MB for the Catalog Server and 64 MB for the StateStore.

For the Impala Daemon memory limit, the minimum is 256 MB and the ideal is (total RAM on daemon host) * 0.64.

Solr

For the Solr Server JVM heap, the minimum is 50 MB and the ideal is min(64 GB, (total RAM on Solr Server host) * 0.64) / 2.6. For the Solr Server JVM direct memory segment, the minimum is 256 MB and the ideal is min(64 GB, (total RAM on Solr Server host) * 0.64) / 2.

Cloudera Management Service

- Alert Publisher JVM heap - Treated as if it consumed a fixed amount of memory by setting the minimum/ideal consumption values, but not the configuration values. The two consumption values are set to 256 MB.
- Service and Host Monitor JVM heaps - The minimum is 50 MB and the ideal is either 256 MB (10 or fewer managed hosts), 1 GB (100 or fewer managed hosts), or 2 GB (over 100 managed hosts).

- Event Server, Reports Manager, and Navigator Audit Server JVM heaps - The minimum is 50 MB and the ideal is 1 GB.
- Navigator Metadata Server JVM heap - The minimum is 512 MB and the ideal is 2 GB.
- Service and Host Monitor off-heap memory segments - The minimum is either 768 MB (10 or fewer managed hosts), 2 GB (100 or fewer managed hosts), or 6 GB (over 100 managed hosts). The ideal is always twice the minimum.

Step 2 Generic Rule for Static Service Pools Wizard

For every {role, segment} pair where the segment defines a default value and an autoconfiguration share, the pair's minimum is set to the segment's default value, and the ideal is set to min((segment soft max (if exists) or segment max (if exists) or 2^63-1), (total RAM on role's host * 0.8 / segment scale factor * service percentage chosen in wizard * segment autoconfiguration share)).

Autoconfiguration shares are defined as follows:

- HBase RegionServer JVM heap: 1
- HDFS DataNode JVM heap: 1 in CDH 4, 0.2 in CDH 5
- HDFS DataNode maximum locked memory: 0.8 (CDH 5 only)
- Solr Server JVM heap: 0.5
- Solr Server JVM direct memory: 0.5
- Spark Standalone Worker JVM heap: 1
- Accumulo Tablet Server JVM heap: 1
- Add-on services: any

Roles not mentioned here do not define autoconfiguration shares and thus aren't affected by this rule.

Additionally, there's a generic rule to handle cgroup.memory_limit_in_bytes, which is unused by Cloudera services but is available for add-on services. Its behavior varies depending on whether the role in question has segments or not.

With Segments

The minimum is the min(cgroup.memory_limit_in_bytes_min (if exists) or 0, sum_over_all(segment minimum consumption)), and the ideal is the sum of all segment ideal consumptions.

Without Segments

The minimum is cgroup.memory_limit_in_bytes_min (if exists) or 0, and the ideal is (total RAM on role's host * 0.8 * service percentage chosen in wizard).

Step 3 Custom Rules for Static Service Pools Wizard

YARN

For the memory total allowed for containers, the minimum is 1 GB and the ideal is min(8 GB, (total RAM on Node Manager host) * 0.8 * service percentage chosen in wizard).

Impala

For the Impala Daemon memory limit, the minimum is 256 MB and the ideal is ((total RAM on Daemon host) * 0.8 * service percentage chosen in wizard).

MapReduce

- Mapper JVM heaps - the minimum is 1 and the ideal is (number of cores, including hyperthreads, on the TaskTracker host * service percentage chosen in wizard). Memory consumption is scaled by mapred_child_java_opts_max_heap (the size of a given task's heap).

- Reducer JVM heaps - the minimum is 1 and the ideal is (number of cores, including hyperthreads on the TaskTracker host * service percentage chosen in wizard) / 2. Memory consumption is scaled by mapred_child_java_opts_max_heap (the size of a given task's heap).

Step 3 Generic Rule

For every {role, segment} pair, the segment's current value is converted into bytes, and then multiplied by the scale factor (1.0 by default, 1.3 for JVM heaps, and freely defined for Custom Service Descriptor services).

Step 3 Custom Rules

Impala

For the Impala Daemon, the memory consumption is 0 if YARN Service for Resource Management is set. If the memory limit is defined but not -1, its value is used verbatim. If it's defined but -1, the consumption is equal to the total RAM on the Daemon host. If it is undefined, the consumption is (total RAM * 0.8).

Solr

For the Solr Server JVM direct memory segment, the consumption is equal to the value verbatim provided solr.hdfs.blockcache.enable and solr.hdfs.blockcache.direct.memory.allocation are both true. Otherwise, the consumption is 0.

Step 7 Custom Rules

HDFS

- NameNode JVM heaps are equalized. For every pair of NameNodes in an HDFS service with different heap sizes, the larger heap size is reset to the smaller one.
- JournalNode JVM heaps are equalized. For every pair of JournalNodes in an HDFS service with different heap sizes, the larger heap size is reset to the smaller one.
- NameNode and Secondary NameNode JVM heaps are equalized. For every {NameNode, Secondary NameNode} pair in an HDFS service with different heap sizes, the larger heap size is reset to the smaller one.

HBase

Master JVM heaps are equalized. For every pair of Masters in an HBase service with different heap sizes, the larger heap size is reset to the smaller one.

Impala

If an Impala service has YARN Service for Resource Management set, every Impala Daemon memory limit is set to the value of (yarn.nodemanager.resource.memory-mb * 1 GB) if there's a YARN NodeManager co-located with the Impala Daemon.

MapReduce

JobTracker JVM heaps are equalized. For every pair of JobTrackers in an MapReduce service with different heap sizes, the larger heap size is reset to the smaller one.

Oozie

Oozie Server JVM heaps are equalized. For every pair of Oozie Servers in an Oozie service with different heap sizes, the larger heap size is reset to the smaller one.

YARN

ResourceManager JVM heaps are equalized. For every pair of ResourceManagers in a YARN service with different heap sizes, the larger heap size is reset to the smaller one.

ZooKeeper

ZooKeeper Server JVM heaps are equalized. For every pair of servers in a ZooKeeper service with different heap sizes, the larger heap size is reset to the smaller one.

General Rules

HBase

- hbase.replication - For each HBase service, set to true if there's a Key-Value Store Indexer service in the cluster. This rule is unscoped; it can fire even if the HBase service is not under scope.
- replication.replicationsource.implementation - For each HBase service, set to com.ngdata.sep.impl.SepReplicationSource if there's a Keystore Indexer service in the cluster. This rule is unscoped; it can fire even if the HBase service is not under scope.

HDFS

- dfs.datanode.du.reserved - For each DataNode, set to min((total space of DataNode host largest mountpoint) / 10, 10 GB).
- dfs.namenode.name.dir - For each NameNode, set to the first two mountpoints on the NameNode host with /dfs/nn appended.
- dfs.namenode.checkpoint.dir - For each Secondary NameNode, set to the first mountpoint on the Secondary NameNode host with /dfs/snn appended.
- dfs.datanode.data.dir - For each DataNode, set to all the mountpoints on the host with /dfs/dn appended.
- dfs.journalnode.edits.dir - For each JournalNode, set to the first mountpoint on the JournalNode host with /dfs/jn appended.
- dfs.datanode.failed.volumes.tolerated - For each DataNode, set to (number of mountpoints on DataNode host) / 2.
- dfs.namenode.service.handler.count and dfs.namenode.handler.count - For each NameNode, set to ln(number of DataNodes in this HDFS service) * 20.
- dfs.datanode.hdfs-blocks-metadata.enabled - For each HDFS service, set to true if there's an Impala service in the cluster. This rule is unscoped; it can fire even if the HDFS service is not under scope.
- dfs.client.read.shortcircuit - For each HDFS service, set to true if there's an Impala service in the cluster. This rule is unscoped; it can fire even if the HDFS service is not under scope.
- dfs.datanode.data.dir.perm - For each DataNode, set to 755 if there's an Impala service in the cluster and the cluster isn't Kerberized. This rule is unscoped; it can fire even if the HDFS service is not under scope.
- fs.trash.interval - For each HDFS service, set to 1.

Hue

- WebHDFS dependency - For each Hue service, set to either the first HttpFS role in the cluster, or, if there are none, the first NameNode in the cluster.
- HBase Thrift Server dependency- For each Hue service in a CDH 4.4 or higher cluster, set to the first HBase Thrift Server in the cluster.

Impala

For each Impala service, set Enable Audit Collection and Enable Lineage Collection to true if there's a Cloudera Management Service with a Navigator Audit Server and Navigator Metadata Server roles. This rule is unscoped; it can fire even if the Impala service is not under scope.

MapReduce

- mapred.local.dir - For each JobTracker, set to the first mountpoint on the JobTracker host with /mapred/jt appended.
- mapred.local.dir - For each TaskTracker, set to all the mountpoints on the host with /mapred/local appended.

- mapred.reduce.tasks - For each MapReduce service, set to max(1, sum_over_all(TaskTracker number of reduce tasks (determined via mapred.tasktracker.reduce.tasks.maximum for that TaskTracker, which is configured s eparately)) / 2).
- mapred.job.tracker.handler.count - For each JobTracker, set to max(10, ln(number of TaskTrackers in this Ma pReduce service) * 20).
- mapred.submit.replication - If there's an HDFS service in the cluster, for each MapReduce service, set to max(min(number of DataNodes in the HDFS service, value of HDFS Replication Factor), sqrt(number of DataNodes in the HDFS service)).
- mapred.tasktracker.instrumentation - If there's a management service, for each MapReduce service, set to org.apac he.hadoop.mapred.TaskTrackerCmonInst. This rule is unscoped; it can fire even if the MapReduce service is not under scope.

YARN

- yarn.nodemanager.local-dirs - For each NodeManager, set to all the mountpoints on the NodeManager host with / yarn/nm appended.
- yarn.nodemanager.resource.cpu-vcores - For each NodeManager, set to the number of cores (including hyperthreads) on the NodeManager host.
- mapred.reduce.tasks - For each YARN service, set to max(1,sum_over_all(NodeManager number of cores, incl uding hyperthreads) / 2).
- yarn.resourcemanager.nodemanagers.heartbeat-interval-ms - For each NodeManager, set to max(100, 10 * (n umber of NodeManagers in this YARN service)).
- yarn.scheduler.maximum-allocation-vcores - For each ResourceManager, set to max_over_all(NodeManager number of vcores (determined via yarn.nodemanager.resource.cpu-vcores for that NodeManager, which is configured separately)).
- yarn.scheduler.maximum-allocation-mb - For each ResourceManager, set to max_over_all(NodeManager amount of RAM (determined via yarn.nodemanager.resource.memory-mb for that NodeManager, which is conf igured separately)).
- mapreduce.client.submit.file.replication - If there's an HDFS service in the cluster, for each YARN service, set to max(min(number of DataNodes in the HDFS service, value of HDFS Replication Factor), sqrt(number of D ataNodes in the HDFS service)).

All Services

If a service dependency is unset, and a service with the desired type exists in the cluster, set the service dependency to the first such target service. Applies to all service dependencies except YARN Service for Resource Management. Applies only to the Installation and Add Cluster wizards.

Role-Host Placement

Cloudera Manager employs the same role-host placement rule regardless of wizard. The set of hosts considered depends on the scope. If the scope is a cluster, all hosts in the cluster are included. If a service, all hosts in the service's cluster are included. If the Cloudera Management Service, all hosts in the deployment are included. The rules are as follows:

1. The hosts are sorted from most to least physical RAM. Ties are broken by sorting on hostname (ascending) followed by host identifier (ascending).
2. The overall number of hosts is used to determine which arrangement to use. These arrangements are hard-coded, each dictating for a given "master" role type, what index (or indexes) into the sorted host list in step 1 to use.

3. Master role types are included based on several factors:
   - Is this role type part of the service (or services) under scope?
   - Does the service already have the right number of instances of this role type?
   - Does the cluster's CDH version support this role type?
   - Does the installed license allow for this role type to exist?
4. Master roles are placed on each host using the indexes and the sorted host list. If a host already has a given master role, it is skipped.
5. An HDFS DataNode is placed on every host outside of the arrangement described in step 2, provided HDFS is one of the services under scope.
6. Certain "worker" roles are placed on every host where an HDFS DataNode exists, either because it existed there prior to the wizard, or because it was added in the previous step. The supported worker role types are:
   - MapReduce TaskTrackers
   - YARN NodeManagers
   - HBase RegionServers
   - Impala Daemons
   - Spark Workers
7. Hive gateways are placed on every host, provided a Hive service is under scope and a gateway didn't already exist on a given host.
8. Spark on YARN gateways are placed on every host, provided a Spark on YARN service is under scope and a gateway didn't already exist on a given host.

This rule merely dictates the default placement of roles; you are free to modify it before it is applied by the wizard.

Using the Cloudera Manager API

The Cloudera Manager API provides configuration and service lifecycle management, service health information and metrics, and allows you to configure Cloudera Manager itself. The API is served on the same host and port as the Cloudera Manager Admin Console, and does not require an extra process or extra configuration. The API supports HTTP Basic Authentication, accepting the same users and credentials as the Cloudera Manager Admin Console.

You can also access the Cloudera Manager Swagger API user interface from the Cloudera Manager Admin Console. Go to SupportAPI Explorer to open Swagger.

Resources

Cloudera Manager REST API documentation

Javadoc

Cloudera Manager API tutorial

Current API version: v43

Using the Cloudera Manager API to backup and restore clusters

Procedures, examples and resources for using the Cloudera Manager API to automate cluster operations.

Backing up the Cloudera Manager configuration

Steps to export the Cloudera Manager configuration.

Before you begin

Minimum Required Role: Cluster Administrator (also provided by Full Administrator) This feature is not available when using Cloudera Manager to manage Data Hub clusters.

By default, Cloudera Manager redacts sensitive information from the exported configuration JSON file. If you need the ability to restore the Cloudera Manager configuration, you must do one of the following:

- Disable redaction. The JSON file will contain all the configurations, including sensitive information and can be used to restore the Cloudera Manager configuration.
- Replace the redacted information. The JSON will contain the word "REDACTED" where sensitive information was redacted. Replace these values with the correct values before restoring the Cloudera Manager configuration.

Procedure

Exporting the Cloudera Manager Configuration

1. (Optional) If you want to export the configuration without redaction, you can disable redaction by setting a JVM parameter:
   a) Log in the Cloudera Manager server host using ssh.
   b) Edit the /etc/default/cloudera-scm-server file by adding the following property (separate each property with a space) to the line that begins with export CMF_JAVA_OPTS.

      ```
      -Dcom.cloudera.api.redaction=false
      ```

      For example:

      ```
      export CMF_JAVA_OPTS="-Xmx2G -Dcom.cloudera.api.redaction=false"
      ```

   c) Restart Cloudera Manager:

      ```
      sudo service cloudera-scm-server restart
      ```

2. Export the Cloudera Manager configuration:
   a) Log in to a host that has network access to the Cloudera Manager server host using ssh.
   b) Run the following command:

      ```
      # curl -u admin_uname:admin_pass "http://cm_server_host:7180/api/v43/cm/
      deployment" >
      path_to_file/cm-deployment.json
      ```

      Where:

      - *admin_uname* is a username with either the Full Administrator or Cluster Administrator role.
      - *admin_pass* is the password for the *admin_uname* username.
      - *cm_server_host* is the hostname of the Cloudera Manager server.
      - *path_to_file* is the path to the file where you want to save the configuration.

Restoring the Cloudera Manager configuration

Steps to import the Cloudera Manager configuration.

Before you begin

Minimum Required Role: Cluster Administrator (also provided by Full Administrator) This feature is not available when using Cloudera Manager to manage Data Hub clusters.

By default, Cloudera Manager redacts sensitive information from the exported configuration JSON file. If you need the ability to restore the Cloudera Manager configuration, you must do one of the following:

- Disable redaction. The JSON file will contain all the configurations, including sensitive information and can be used to restore the Cloudera Manager configuration.
- Replace the redacted information. The JSON will contain the word "REDACTED" where sensitive information was redacted. Replace these values with the correct values before restoring the Cloudera Manager configuration.

Procedure

1. (Optional) If you are restoring the Cloudera Manager configuration from a redacted backup JSON file, you must replace all entries with the word "REDACTED" with the appropriate value.
2. Restore the Cloudera Manager Configuration

   Using a previously saved JSON document that contains the Cloudera Manager configuration data, you can restore that configuration to a running cluster.

   a) Using the Cloudera Manager Administration Console, stop all running services in your cluster:

      1. On the HomeStatus tab, click  to the right of the cluster name and select Stop.
      2. Click Stop in the confirmation screen. The Command Details window shows the progress of stopping services.

         When All services successfully stopped appears, the task is complete and you can close the Command Details window.

      Warning: If you do not stop the cluster before making this API call, the API call will stop all cluster services before running the job. Any running jobs and data are lost.

   b) Log in to a host that has network access to the Cloudera Manager server host using ssh.
   c) Copy the exported JSON file to this host.
   d) Run the following command:

   ```
   curl -H "Content-Type: application/json" --upload-file path_to_file/cm-deployment.json -u admin:admin http://cm_server_host:7180/api/v43/cm/deployment?deleteCurrentDeployment=true
   ```

   Where:

   - *admin_uname* is a username with either the Full Administrator or Cluster Administrator role.
   - *admin_pass* is the password for the *admin_uname* username.
   - *cm_server_host* is the hostname of the Cloudera Manager server.
   - *path_to_file* is the path to the file containing the JSON configuration file.

Using the Cloudera Manager API to Manage and Configure Clusters

Procedures, examples and resources for using the Cloudera Manager API to automate cluster operations.

Using the Cloudera Manager API for Cluster Automation

How to use the Cloudera Manager API to automate cluster management.

One of the complexities of Apache Hadoop is the need to deploy clusters of servers, potentially on a regular basis. If you maintain hundreds of test and development clusters in different configurations, this process can be complex and cumbersome if it is not automated.

Cluster Automation Use Cases

Cluster automation is useful in various situations. For example, you might work on many versions of Cloudera Runtime, which works on a wide variety of OS distributions (RHEL, Ubuntu, and SLES). You might have complex configuration combinations—highly available HDFS or simple HDFS, Kerberized or non-secure, YARN or MRv1,

and so on. With these requirements, you need an easy way to create a new cluster that has the required setup. This cluster can also be used for integration, testing, customer support, demonstrations, and other purposes.

You can install and configure components according to precise specifications using the Cloudera Manager REST API. Using the API, you can add hosts, install Cloudera Runtime, and define the cluster and its services. You can also tune heap sizes, set up HDFS HA, turn on Kerberos security and generate keytabs, and customize service directories and ports. Every configuration available in Cloudera Manager is exposed in the API.

The API also provides access to management functions:

- Obtaining logs and monitoring the system
- Starting and stopping services
- Polling cluster events
- Creating a disaster recovery replication schedule

For example, you can use the API to retrieve logs from HDFS, HBase, or any other service, without knowing the log locations. You can also stop any service with no additional steps.

Use cases for the Cloudera Manager API for cluster automation might include:

- OEM and hardware partners that deliver Hadoop-in-a-box appliances using the API to set up Cloudera Runtime and Cloudera Manager on bare metal in the factory.
- Automated deployment of new clusters, using a combination of Puppet and the Cloudera Manager API. Puppet does the OS-level provisioning and installs the software. The Cloudera Manager API sets up the Hadoop services and configures the cluster.
- Integrating the API with reporting and alerting infrastructure. An external script can poll the API for health and metrics information, as well as the stream of events and alerts, to feed into a custom dashboard.

Java API Example

This example covers the Java API client.

To use the Java client, add this dependency to your project's pom.xml:

```
<project>
  <repositories>
    <repository>
      <id>cdh.repo</id>
      <url>https://repository.cloudera.com/artifactory/cloudera-repos</url>
      <name>Cloudera Repository</name>
    </repository>
    …
  </repositories>
  <dependencies>
    <dependency>
      <groupId>com.cloudera.api</groupId>
      <artifactId>cloudera-manager-api</artifactId>
      <version>4.6.2</version>      <!-- Set to the version of Cloudera Man
ager you use -->
    </dependency>
    …
  </dependencies>
  ...
</project>
```

The Java client works like a proxy. It hides from the caller any details about REST, HTTP, and JSON. The entry point is a handle to the root of the API:

```
RootResourcev43 apiRoot = new ClouderaManagerClientBuilder().withHost("cm.c
loudera.com")
.withUsernamePassword("admin", "admin").build().getRootv43();
```

From the root, you can traverse down to all other resources. (It's called "v43" because that is the current Cloudera Manager API version, but the same builder will also return a root from an earlier version of the API.) The tree view shows some key resources and supported operations:

- RootResourcev43
  - ClustersResourcev43 - host membership, start cluster
    - ServicesResourcev43 - configuration, get metrics, HA, service commands
      - RolesResource - add roles, get metrics, logs
      - RoleConfigGroupsResource - configuration
    - ParcelsResource - parcel management
- HostsResource - host management, get metrics
- UsersResource - user management

For more information, see the Javadoc.

The following example lists and starts a cluster:

```
// List of clusters
ApiClusterList clusters = apiRoot.getClustersResource().readClusters(DataVie
w.SUMMARY);
for (ApiCluster cluster : clusters) {
  LOG.info("{}: {}", cluster.getName(), cluster.getVersion());
}

// Start the first cluster
ApiCommand cmd = apiRoot.getClustersResource().startCommand(clusters.get(
0).getName());
while (cmd.isActive()) {
   Thread.sleep(100);
   cmd = apiRoot.getCommandsResource().readCommand(cmd.getId());
}
LOG.info("Cluster start {}", cmd.getSuccess() ? "succeeded" : "failed " + c
md.getResultMessage());
```

Python Example

You can see an example of automation with Python at the following link: Python example. The example contains information on the requirements and steps to automate a cluster deployment.

Using the Cloudera Manager API to Obtain Configuration Files

You can use the Cloudera Manager API to obtain configuration files.

About this task

Procedure

1. Obtain the list of a service's roles:

   ```
   http://cm_server_host:7180/api/v43/clusters/clusterName/service
   s/serviceName/roles
   ```

2. Obtain the list of configuration files a process is using:

   ```
   http://cm_server_host:7180/api/v43/clusters/clusterName/service
   s/serviceName/roles/roleName/process
   ```

3. Obtain the content of any particular file:

```
http://cm_server_host:7180/api/v43/clusters/clusterName/service
s/serviceName/roles/roleName/process/
configFiles/configFileName
```

For example:

```
http://cm_server_host:7180/api/v43/clusters/Cluster%201/services/OOZIE-1/r
oles/
OOZIE-1-OOZIE_SERVER-e121641328fcb107999f2b5fd856880d/process/configFiles/
oozie-site.xml
```

Retrieving Service and Host Properties

To update a service property using the Cloudera Manager APIs, you will need to know the name of the property, not just the display name. If you know the property's display name but not the property name itself, retrieve the documentation by requesting any configuration object with the query string view=FULL appended to the URL. For example:

```
http://cm_server_host:7180/api/v43/clusters/Cluster%201/servic
es/service_name/config?view=FULL
```

Search the results for the display name of the desired property. For example, a search for the display name HDFS Service Environment Advanced Configuration Snippet (Safety Valve) shows that the corresponding property name is hdfs_service_env_safety_valve:

```
{
   "name" : "hdfs_service_env_safety_valve",
   "require" : false,
   "displayName" : "HDFS Service Environment Advanced Configuration Snippet
 (Safety Valve)",
   "description" : "For advanced use onlyu, key/value pairs (one on each l
ine) to be inserted into a roles
   environment. Applies to configurations of all roles in this service exce
pt client configuration.",
   "relatedName" : "",
   "validationState" : "OK"
}
```

Similar to finding service properties, you can also find host properties. First, get the host IDs for a cluster with the URL:

```
http://cm_server_host:7180/api/v43/hosts
```

This should return host objects of the form:

```
{
   "hostId" : "2c2e951c-aaf2-4780-a69f-0382181f1821",
   "ipAddress" : "10.30.195.116",
   "hostname" : "cm_server_host",
   "rackId" : "/default",
   "hostUrl" : "http://cm_server_host:7180/cmf/hostRedirect/2c2e951c-adf2
-4780-a69f-0382181f1821",
   "maintenanceMode" : false,
   "maintenanceOwners" : [ ],
   "commissionState" : "COMMISSIONED",
   "numCores" : 4,
   "totalPhysMemBytes" : 10371174400
}
```

Then obtain the host properties by including one of the returned host IDs in the URL:

```
http://cm_server_host:7180/api/v43/hosts/2c2e951c-adf2-4780-a69f-0382181f1
821?view=FULL
```

Using the Cloudera Manager API to Set Advanced Configuration Snippets (Safety Valves)

You can use the Cloudera Manager API to set service configuration properties.

Before you begin

You must have network access to the Cloudera Manager Admin Console.

Procedure

1. Find the name of the configuration property.
   a) Go to the Cloudera Manager Configuration Property Reference and navigate to the page for the Cloudera Runtime version and service you want to configure.
   b) Search through the page for the Advanced Configuration Snippet property.
   c) Copy the value of the API Name.
2. Determine whether the Advanced Configuration Snippet property is defined using name/value pairs or XML:
   a) Log in to the Cloudera Manager Admin Console.
   b) Go to the service where you want to change the advanced configuration.
   c) Click the Configuration tab.
   d) Search for the Advanced Configuration Snippet property.
      - If the property is a text file using name/value pairs, the description includes a View as text link.
      - If the property is an XML file, the description includes a View as XML link.
3. Log into a host with network access to the Cloudera Manager Admin Console using ssh, and run one of the following commands:
   - Snippet is defined using name/value pairs:

     ```
     curl -X PUT -u username:password
      "Cloudera_Manager_server_name:Cloudera_Manager_server_port/api/v43/clu
     sters/Cluster_name/services/service_name/config" -H  "accept: applicatio
     n/json" -H  "Content-Type: application/json" -d "{  \"items\": [
     {      \"name\": \"API-Name\",      \"value\": \"name=value\",      \"s
     ensitive\": false    }  ]}"
     ```

     For example, the following command changes the value of the HDFS Replication Environment Advanced Configuration Snippet (Safety Valve) in the HDFS-1 service in Cluster 1 to test=2:
   - ```
     curl -X PUT -u admin:admin "http://myCMserver.com:7180/api/v43/cluster
     s/Cluster%201/services/HDFS-1/config" -H  "accept: application/json" -H
      "Content-Type: application/json" -d "{  \"items\": [        {      \"na
     ```
     ```

```
me\": \"hdfs_replication_env_safety_valve\",      \"value\": \"test=2\",
      \"sensitive\": false    }  ]}"
```

Important: The value you enter overwrites all the contents of the Snippet, so you must also include any existing values. For example, if only the following value is currently configured: foo1=bar1, then to add an additional name/value pair, you must set the value like this:

```
\"value\": \"foo1=bar1 foo2=bar2\"
```

This command sets the Advanced Configuration Snippet to the following:

```
foo1=bar1
foo2=bar2
```

- Snippet is defined using XML:

```
curl -X PUT -u username:password
 "Cloudera_Manager_server_name:Cloudera_Manager_server_port/api/v43/clu
sters/Cluster_name/services/service_name/config" -H  "accept: applicatio
n/json" -H  "Content-Type: application/json" -d "{  \"items\": [
{      \"name\": \"API-Name\",      \"value\": \"XML structure\",
 \"sensitive\": false    }  ]}"
```

For example:

```
curl -X PUT -u admin:admin "http://myCMserver.com:7180/api/v43/cluster
s/Cluster%201/services/HDFS-1/config" -H  "accept: application/json" -H
 "Content-Type: application/json" -d "{  \"items\": [        {      \"na
me\": \"hdfs_replication_env_safety_valve\",      \"value\": \"<property
><name>foo</name><value>bar</value><description>Example</description></p
roperty>\",      \"sensitive\": false    }  ]}"
```

This call sets the Advanced Configuration Snippet to the following:

```
<property>
  <name>foo</name>
  <value>bar</value>
  <description>Example</description>
</property>
```

- Important: The value you enter overwrites all the contents of the Snippet, so you must also include any previously-existing <property> elements.

Results

The configuration value has been changed.

Related Information

Cloudera Manager Configuration Properties Reference

Using Tags in Cloudera Manager

Tags provide a way to add information to clusters, hosts, services, and roles using the Cloudera Manager API.

A tag in the context of Cloudera Manager is a name-value pair that is attached to a cluster, service, role, or host. Tags allow information to be added to an entity in a very open-ended fashion. A single entity may have any number of tags, as long as each tag name is unique, and tags may be added or removed at any time. All tags appear in diagnostic bundles.

Tags may be created by various cluster components, using the following naming convention: _cldr_{component_specific_prefix}_tagname. You can also use the Cloudera Manager API to create, update, delete, and retrieve tags.

Setting and Retrieving Tags using the Cloudera Manager API

A TagsResource endpoint for retrieving tags globally is available in versions 41 and higher of the Cloudera Manager API. Endpoints to read, delete, and add or modify tags on clusters, services, hosts and roles are also available. For complete documentation, open the Cloudera Manager Admin Console and go to Support API Explorer.

You can manage tags for services, roles, and hosts using the following endpoints:

Clusters

/clusters/*{clusterName}*/tags

Hosts

/hosts/*{hostname}*/tags

Services

/clusters/*{clusterName}*/services/<service-name>/tags

Roles

/clusters/*{clusterName}*/services/*{serviceName}*/roles/*{roleName}*/tags

Retrieve all tags:

/tags

API Examples

The following examples show how to add, modify, or delete tags for a cluster:

Add or update a tag on a cluster

This example adds a tag with the name example_name and value example_value to Cluster1. if a tag with the name example_name is already attached to Cluster1, the value of that tag will be updated to *example_value*.

```
curl -X PUT --header 'Content-Type: application/json' --header '
Accept: application/json' -d '[
  {
    "name": "example_name",
    "value": "example_value"
  }
]' 'http://username:password@<Cloudera Manager_URL>:7180/api/v43/
clusters/<cluster-name>/tags'
```

Delete a tag on a cluster

This example deletes the tag named example_name from Cluster1, if the tag exists.

```
curl -X DELETE --header 'Content-Type: application/json' --header
 'Accept: application/json' -d '[
  {
    "name": "example_name"
  }
]' 'http://username:password@<Cloudera Manager_URL>l:7180/a
pi/v43/clusters/<cluster-name>/tags'
```

Retrieve all tags

This example returns a serialized JSON list of all tags, including which entities each is attached to, with a maximum of 10 tags and beginning with the most recently created tag.

```
curl -X GET --header 'Accept: application/json' http
://username:password@<Cloudera Manager_URL>:7180/api/v43/tags?li
mit=10&offset=0'
```

Setting Tags using ImportClusterTemplate (Example)

You can set tags on clusters, services, and hosts (but not roles) using a cluster template provided to /cm/importCluste rTemplate in the Cloudera Manager API versions 41 and higher.

The following is an example of a template using tags:

```
{
  "cdhVersion" : "7.0.2",
  "displayName" : "Cluster 2",
  "cmVersion" : "7.0.2",
  "tags" : [ {
 "name" : "_cldr_cm_clustertag",
 "value" : "example1"
  }],
  "services" : [ {
    "refName" : "hdfs",
    "serviceType" : "HDFS",
    "tags" : [ {
   "name" : "_cldr_cm_servicetag",
   "value" : "example2"
    }],
    "roleConfigGroups" : [ {
      "refName" : "hdfs-NAMENODE-BASE",
      "roleType" : "NAMENODE",
      "base" : true
    },
{
      "refName" : "hdfs-DATANODE-BASE",
      "roleType" : "DATANODE",
      "base" : true
  } ]
} ],
  "hostTemplates" : [ {
    "refName" : "HostTemplate1",
    "cardinality" : 1,
    "roleConfigGroupsRefNames" : [ "hdfs-NAMENODE-BASE" ]
    "tags" : [ {
   "name" : "_cldr_cm_hosttag",
   "value" : "example3"
  }],
  } ,{
    "refName" : "HostTemplate2",
    "cardinality" : 1,
    "roleConfigGroupsRefNames" : [ "hdfs-DATANODE-BASE" ]
  } ],
  "instantiator" : {
    "clusterName" : "Cluster 2",
    "hosts" : [ {
      "hostName" : "host00003",
      "hostTemplateRefName" : "HostTemplate1"
    },
 {
      "hostName" : "host00004",
      "hostTemplateRefName" : "HostTemplate2"
    }],
    "lenient" : false
  }
}
```

This adds the _cldr_cm_clustertag to the cluster that is being created, the _cldr_cm_servicetag to the HDFS service on that cluster, and the _cldr_cm_hosttag to host00003. Note that even if the host templates are persisted (which is

optional during cluster template import), the _cldr_cm_hosttag will no longer be associated with HostTemplate1, only with host00003 itself.

Initiating HDFS failover using the Cloudera Manager API

Steps to initiate failover of HDFS using the Cloudera Manager API.

Before you begin

The cluster must have HDFS High Availability enabled.

Procedure

1. Determine the ID of the Active and Standby NameNode roles. Run the following command:

```
curl -X GET "http://myCluster-1:7180/api/v<API VERSION>/cluster
s/cluster_name/services/HDFS_service_name/roles?view=summary" -H "accept:
 application/json"
```

For example:

```
curl -X GET "http://myCluster-1:7180/api/v43/clusters/Cluster%201/servic
es/HDFS-1/roles?view=summary" -H "accept: application/json"
```

A data structure is returned that contains the NameNode IDs. Look for output similar to the following:

```
      "name": "HDFS-1-NAMENODE-b520c25b659296aea5c8249c96a41b79",
      "type": "NAMENODE",
      "serviceRef": {
        "clusterName": "Cluster 1",
        "serviceName": "HDFS-1",
        "serviceDisplayName": "HDFS-1",
        "serviceType": "HDFS"
      },
      "hostRef": {
        "hostId": "dd503ff5-4c27-4261-b0f1-d9273ad6b510",
        "hostname": "nightly7x-cmha-4.nightly7x-cmha.root.hwx.site"
      },
      "roleUrl": "http://nightly7x-cmha-4.nightly7x-cmha.root.hwx.site:71
80/cmf/roleRedirect/HDFS-1-NAMENODE-b520c25b659296aea5c8249c96a41b79",
      "roleState": "STARTED",
      "healthSummary": "GOOD",
      "configStalenessStatus": "FRESH",
      "haStatus": "ACTIVE",
      "maintenanceMode": false,
      "commissionState": "COMMISSIONED",
      "roleConfigGroupRef": {
        "roleConfigGroupName": "HDFS-1-NAMENODE-BASE"
      },
```

In the response above, the line "haStatus": "ACTIVE" indicates that this block defines the Active NameNode. The NameNode ID is the value of the "name" property, in this case the ID is HDFS-1-NAMENODE-b520c25b6592 96aea5c8249c96a41b79. You will use this value in the next step.

```
      "name": "HDFS-1-NAMENODE-f9c734734bb24fb4af93576ebeb13ad9",
      "type": "NAMENODE",
      "serviceRef": {
        "clusterName": "Cluster 1",
        "serviceName": "HDFS-1",
        "serviceDisplayName": "HDFS-1",
        "serviceType": "HDFS"
      },
```

```
      "hostRef": {
        "hostId": "5dc7ce94-9db0-4a50-bd82-f0bb64da624b",
        "hostname": "nightly7x-cmha-1.nightly7x-cmha.root.hwx.site"
      },
      "roleUrl": "http://nightly7x-cmha-4.nightly7x-cmha.root.hwx.site:718
0/cmf/roleRedirect/HDFS-1-NAMENODE-f9c734734bb24fb4af93576ebeb13ad9",
      "roleState": "STARTED",
      "healthSummary": "GOOD",
      "configStalenessStatus": "FRESH",
      "haStatus": "STANDBY",
      "maintenanceMode": false,
      "commissionState": "COMMISSIONED",
      "roleConfigGroupRef": {
        "roleConfigGroupName": "HDFS-1-NAMENODE-BASE"
      },
```

In the response above, the line "haStatus": "STANDBY" indicates that this block defines the Standby NameNode. The NameNode ID is the value of the "name" property, in this case the ID is HDFS-1-NAMENODE-f9c73473 4bb24fb4af93576ebeb13ad9. You will use this value in the next step.

2. Run the following command to initiate the HDFS failover, substituting the NameNode IDs of the Active and Standby NameNodes :

```
 curl -X POST "Cloudera_Manager_hostname:Cloudera_Manager_port/api/v<API
 VERSION>/clusters/cluster_name/services/HDFS_service_name/commands/hdfsFa
ilover?force=true|false" -H "accept: application/json" -H "Content-Type
: application/json" -d "{ \"items\": [ \"Active_NameNode_ID\", \"Standby
_NameNode_ID\" ]}"
```

For example:

```
curl -X POST “http://myCluster-1.com:7180/api/v43/clusters/Cluster%201/s
ervices/HDFS-1/commands/hdfsFailover?force=true” -H “accept: application
/json” -H “Content-Type: application/json” -d “{ \“items\“: [ \“HDFS-1-N
AMENODE-b520c25b659296aea5c8249c96a41b79\“, \“HDFS-1-NAMENODE-f9c734734b
b24fb4af93576ebeb13ad9\” ]}”
```

Related Information

High Availability on HDFS clusters

Creating a Runtime Cluster Using a Cloudera Manager Template

You can create a new cluster by exporting a cluster template from an existing cluster managed by Cloudera Manager. You can then modify the template and use it to create new clusters with the same configuration on a new set of hosts.

About this task

Use cluster templates to:

- Duplicate clusters for use in developer, test, and production environments.
- Quickly create a cluster for a specific workload.
- Reproduce a production cluster for testing and debugging.

Follow these general steps and refer to the following tasks to create a template and a new cluster:

Before you begin

You can only import a cluster template to create a new cluster that is running the same version of Cloudera Runtime as the cluster from which the template was exported.

Procedure

1. Export the cluster configuration from the source cluster. The exported configuration is a JSON file that details all of the configurations of the cluster. The JSON file includes an instantiator section that contains some values you must provide before creating the new cluster.
2. Set up the hosts for the new cluster by installing Cloudera Manager agents and the JDK on all hosts. For secure clusters, also configure a Kerberos key distribution center (KDC) in Cloudera Manager.
3. Create any local repositories required for the cluster.

   See *Step 1: Configure a Repository for Cloudera Manager*.
4. Complete the instantiator section of the cluster configuration JSON document to create a template.
5. Import the cluster template to the new cluster.

Related Information

Configuring a Local Package Repository

Exporting the Cluster Configuration

To create a cluster template, you begin by exporting the configuration from the source cluster. The cluster must be running and managed by Cloudera Manager.

Before you begin

Minimum Required Role: Cluster Administrator (also provided by Full Administrator) This feature is not available when using Cloudera Manager to manage Data Hub clusters.

By default, Cloudera Manager redacts sensitive information from the exported configuration JSON file. If you need the ability to restore the cluster configuration, you must do one of the following:

- Disable redaction. The JSON file will contain all the configurations, including sensitive information and can be used to restore the cluster configuration.
- Replace the redacted information. The JSON will contain the word "REDACTED" where sensitive information was redacted. Replace these values with the correct values before restoring the cluster configuration.

Procedure

1. (Optional) If you want to export the configuration without redaction, you can disable redaction by setting a JVM parameter:
   a) Log in the Cloudera Manager server host using ssh.
   b) Edit the /etc/default/cloudera-scm-server file by adding the following property (separate each property with a space) to the line that begins with export CMF_JAVA_OPTS.

      ```
      -Dcom.cloudera.api.redaction=false
      ```

      For example:

      ```
      export CMF_JAVA_OPTS="-Xmx2G -Dcom.cloudera.api.redaction=false"
      ```

   c) Restart Cloudera Manager:

      ```
      sudo service cloudera-scm-server restart
      ```
2. Any host templates you have created are used to export the configuration. If you do not want to use those templates in the new cluster, delete them. In Cloudera Manager, go to HostsHost Templates and click Delete next to the Host Template you want to delete.
3. Delete any Host Templates created by the Cloudera Manager Installation Wizard. They typically have a name like Template - 1).

4. Run the following command to download the JSON configuration file to a convenient location for editing:

```
curl -u admin_username:admin_user_password
"http://Cloudera Manager URL/api/v43/clusters/Cluster name/export" >
path_to_file/file_name.json
```

For example:

```
curl -u adminuser:adminpass "http://myCluster-1.myDomain.com:7180/api/v43/
clusters/Cluster1/export" > myCluster1-template.json
```

Note: Add the ?exportAutoConfig=true parameter to the command above to include configurations made by Autoconfiguration. These configurations are included for reference only and are not used when you import the template into a new cluster. For example:

```
curl -u admin_username:admin_user_password
"http://Cloudera Manager URL/api/v43/clusters/Cluster name/export?
exportAutoConfig=true" >
path_to_file/file_name.json
```

Note: 'The cluster from which you export the JSON configuration file may have been installed using either parcels or packages. You can determine the type of installation from the JSON configuration file by examining the first section. If there are entries for "repositories": [], or "products": [], , then the cluster was installed using parcels. Clusters installed using packages do not have these entries.

For example:

```
{
  "cdhVersion" : "5.15.0",
  "displayName" : "Cluster 1",
  "cmVersion" : "5.15.0",
  "repositories" : [ "http://my_cluster.com/cdh5/5.x/parcels", "htt
p://my_cluster.com/cdh5/parcels/ ..."
  "products" : [ {
    "version" : "5.15.0-1.cdh5.15.0.p0.727055",
    "product" : "CDH"
  } ],
```

Preparing a New Cluster

The new cluster into which you import the cluster template must meet the following requirements:

- Database for Cloudera Manager is installed and configured.
- Cloudera Manager is installed and running.
- All required databases for Runtime services are installed. See *Step 4: Install and Configure Databases*.
- The JDK is installed on all cluster hosts.
- The Cloudera Manager Agent is installed and configured on all cluster hosts.
- If the source cluster uses Kerberos, the new cluster must have KDC properties and privileges configured in Cloudera Manager.

Related Information

Install and Configure Databases

Creating the Template

To create a template, modify the instantiator section of the JSON file you downloaded. Lines that contain the string <changeme> require a value that you must supply.

Before you begin

You can only import a cluster template to create a new cluster that is running the same version of Cloudera Runtime as the cluster from which the template was exported.

About this task

Here is a sample instantiator section:

```
  "instantiator" : {
    "clusterName" : "<changeme>",
    "hosts" : [ {
      "hostName" : "<changeme>",
      "hostTemplateRefName" : "<changeme>",
      "roleRefNames" : [ "HDFS-1-NAMENODE-0be88b55f5dedbf7bc74d61a86c0253
e" ]
    }, {
      "hostName" : "<changeme>",
      "hostTemplateRefName" : "<changeme>"
    }, {
      "hostNameRange" : "<HOST[0001-0002]>",
      "hostTemplateRefName" : "<changeme>"
    } ],
    "variables" : [ {
      "name" : "HDFS-1-NAMENODE-BASE-dfs_name_dir_list",
      "value" : "/dfs/nn"
    }, {
      "name" : "HDFS-1-SECONDARYNAMENODE-BASE-fs_checkpoint_dir_list",
      "value" : "/dfs/snn"
    }, {
      "name" : "HIVE-1-hive_metastore_database_host",
      "value" : "myCluster-1.myDomain.com"
    }, {
      "name" : "HIVE-1-hive_metastore_database_name",
      "value" : "hive1"
    }, {
      "name" : "HIVE-1-hive_metastore_database_password",
      "value" : "<changeme>"
    }, {
      "name" : "HIVE-1-hive_metastore_database_port",
      "value" : "3306"
    }, {
      "name" : "HIVE-1-hive_metastore_database_type",
      "value" : "mysql"
    }, {
      "name" : "HIVE-1-hive_metastore_database_user",
      "value" : "hive1"
    }, {
      "name" : "HUE-1-database_host",
      "value" : "myCluster-1.myDomain.com"
    }, {
      "name" : "HUE-1-database_name",
      "value" : "hueserver0be88b55f5dedbf7bc74d61a86c0253e"
    }, {
      "name" : "HUE-1-database_password",
      "value" : "<changeme>"
    }, {
      "name" : "HUE-1-database_port",
      "value" : "3306"
    }, {
      "name" : "HUE-1-database_type",
      "value" : "mysql"
    }, {
      "name" : "HUE-1-database_user",
```

```
      "value" : "hueserver0be88b5"
    }, {
      "name" : "IMPALA-1-IMPALAD-BASE-scratch_dirs",
      "value" : "/impala/impalad"
    }, {
      "name" : "KUDU-1-KUDU_MASTER-BASE-fs_data_dirs",
      "value" : "/var/lib/kudu/master"
    }, {
      "name" : "KUDU-1-KUDU_MASTER-BASE-fs_wal_dir",
      "value" : "/var/lib/kudu/master"
    }, {
      "name" : "KUDU-1-KUDU_TSERVER-BASE-fs_data_dirs",
      "value" : "/var/lib/kudu/tserver"
    }, {
      "name" : "KUDU-1-KUDU_TSERVER-BASE-fs_wal_dir",
      "value" : "/var/lib/kudu/tserver"
    }, {
      "name" : "MAPREDUCE-1-JOBTRACKER-BASE-jobtracker_mapred_local_dir_l
ist",
      "value" : "/mapred/jt"
    }, {
      "name" : "MAPREDUCE-1-TASKTRACKER-BASE-tasktracker_mapred_local_dir_li
st",
      "value" : "/mapred/local"
    }, {
      "name" : "OOZIE-1-OOZIE_SERVER-BASE-oozie_database_host",
      "value" : "myCluster-1.myDomain.com:3306"
    }, {
      "name" : "OOZIE-1-OOZIE_SERVER-BASE-oozie_database_name",
      "value" : "oozieserver0be88b55f5dedbf7bc74d61a86c0253e"
    }, {
      "name" : "OOZIE-1-OOZIE_SERVER-BASE-oozie_database_password",
      "value" : "<changeme>"
    }, {
      "name" : "OOZIE-1-OOZIE_SERVER-BASE-oozie_database_type",
      "value" : "mysql"
    }, {
      "name" : "OOZIE-1-OOZIE_SERVER-BASE-oozie_database_user",
      "value" : "oozieserver0be88"
    }, {
      "name" : "YARN-1-NODEMANAGER-BASE-yarn_nodemanager_local_dirs",
      "value" : "/yarn/nm"
    }, {
      "name" : "YARN-1-NODEMANAGER-BASE-yarn_nodemanager_log_dirs",
      "value" : "/yarn/container-logs"
    } ]
  }
```

Procedure

1. To modify the template, update the hosts section.

   If you have host templates defined in the source cluster, they appear in the hostTemplates section of the JSON template. For hosts that do not use host templates, the export process creates host templates based on role assignments to facilitate creating the new cluster. In either case, you must match the items in the hostTemplates section with the hosts sections in the instantiator section.

   Here is a sample of the hostTemplates section from the same JSON file as the instantiator section, above:

   ```
   "hostTemplates" : [ {
      "refName" : "HostTemplate-0-from-myCluster-1.myDomain.com",
      "cardinality" : 1,
   ```

```
    "roleConfigGroupsRefNames" : [ "FLUME-1-AGENT-BASE", "HBASE-1-GATEWAY-
BASE", "HBASE-1-HBASETHRIFTSERVER-BASE", "HBASE-1-MASTER-BASE", "HDFS-1-
BALANCER-BASE", "HDFS-1-GATEWAY-BASE", "HDFS-1-NAMENODE-BASE", "HDFS-1-N
FSGATEWAY-BASE", "HDFS-1-SECONDARYNAMENODE-BASE", "HIVE-1-GATEWAY-BASE",
 "HIVE-1-HIVEMETASTORE-BASE", "HIVE-1-HIVESERVER2-BASE", "HUE-1-HUE_LOAD
_BALANCER-BASE", "HUE-1-HUE_SERVER-BASE", "IMPALA-1-CATALOGSERVER-BASE",
 "IMPALA-1-STATESTORE-BASE", "KAFKA-1-KAFKA_BROKER-BASE", "KS_INDEXER-1-
HBASE_INDEXER-BASE", "KUDU-1-KUDU_MASTER-BASE", "MAPREDUCE-1-GATEWAY-BAS
E", "MAPREDUCE-1-JOBTRACKER-BASE", "OOZIE-1-OOZIE_SERVER-BASE", "SOLR-1-
SOLR_SERVER-BASE", "SPARK_ON_YARN-1-GATEWAY-BASE", "SPARK_ON_YARN-1-SPAR
K_YARN_HISTORY_SERVER-BASE", "SQOOP-1-SQOOP_SERVER-BASE", "SQOOP_CLIENT-1-
GATEWAY-BASE", "YARN-1-GATEWAY-BASE", "YARN-1-JOBHISTORY-BASE", "YARN-1-RE
SOURCEMANAGER-BASE", "ZOOKEEPER-1-SERVER-BASE" ]
  }, {
    "refName" : "HostTemplate-1-from-myCluster-4.myDomain.com",
    "cardinality" : 1,
    "roleConfigGroupsRefNames" : [ "FLUME-1-AGENT-BASE", "HBASE-1-REGIONS
ERVER-BASE", "HDFS-1-DATANODE-BASE", "HIVE-1-GATEWAY-BASE", "IMPALA-1-IM
PALAD-BASE", "KUDU-1-KUDU_TSERVER-BASE", "MAPREDUCE-1-TASKTRACKER-BASE",
 "SPARK_ON_YARN-1-GATEWAY-BASE", "SQOOP_CLIENT-1-GATEWAY-BASE", "YARN-1-
NODEMANAGER-BASE" ]
  }, {
    "refName" : "HostTemplate-2-from-myCluster-[2-3].myDomain.com",
    "cardinality" : 2,
    "roleConfigGroupsRefNames" : [ "FLUME-1-AGENT-BASE", "HBASE-1-REGIONSE
RVER-BASE", "HDFS-1-DATANODE-BASE", "HIVE-1-GATEWAY-BASE", "IMPALA-1-IMP
ALAD-BASE", "KAFKA-1-KAFKA_BROKER-BASE", "KUDU-1-KUDU_TSERVER-BASE", "MA
PREDUCE-1-TASKTRACKER-BASE", "SPARK_ON_YARN-1-GATEWAY-BASE", "SQOOP_CLIE
NT-1-GATEWAY-BASE", "YARN-1-NODEMANAGER-BASE" ]
  } ]
```

The value of cardinality indicates how many hosts are assigned to the host template in the source cluster.

The value of roleConfigGroupsRefNames indicates which role groups are assigned to the host(s).

Do the following for each host template in the hostTemplates section:

a) Locate the entry in the hosts section of the instantiator where you want the roles to be installed.
b) Copy the value of the refName to the value for hostTemplateRefName.
c) Enter the hostname in the new cluster as the value for hostName. Some host sections might instead use host NameRange for clusters with multiple hosts that have the same set of roles. Indicate a range of hosts by using one of the following:

   - Brackets; for example, myhost[1-4].foo.com
   - A comma-delimited string of hostnames; for example, host-1.domain, host-2.domain, host-3.domain

   Here is an example of the hostTemplates and the hosts section of the instantiator completed correctly:

```
"hostTemplates" : [ {
    "refName" : "HostTemplate-0-from-myCluster-1.myDomain.com",
    "cardinality" : 1,
    "roleConfigGroupsRefNames" : [ "FLUME-1-AGENT-BASE", "HBASE-1-GATEW
AY-BASE", "HBASE-1-HBASETHRIFTSERVER-BASE", "HBASE-1-MASTER-BASE", "HDFS
-1-BALANCER-BASE", "HDFS-1-GATEWAY-BASE", "HDFS-1-NAMENODE-BASE", "HDFS-
1-NFSGATEWAY-BASE", "HDFS-1-SECONDARYNAMENODE-BASE", "HIVE-1-GATEWAY-BAS
E", "HIVE-1-HIVEMETASTORE-BASE", "HIVE-1-HIVESERVER2-BASE", "HUE-1-HUE_L
OAD_BALANCER-BASE", "HUE-1-HUE_SERVER-BASE", "IMPALA-1-CATALOGSERVER-BAS
E", "IMPALA-1-STATESTORE-BASE", "KAFKA-1-KAFKA_BROKER-BASE", "KS_INDEXER
-1-HBASE_INDEXER-BASE", "KUDU-1-KUDU_MASTER-BASE", "MAPREDUCE-1-GATEWAY-
BASE", "MAPREDUCE-1-JOBTRACKER-BASE", "OOZIE-1-OOZIE_SERVER-BASE", "SOLR
-1-SOLR_SERVER-BASE", "SPARK_ON_YARN-1-GATEWAY-BASE", "SPARK_ON_YARN-1-S
PARK_YARN_HISTORY_SERVER-BASE", "SQOOP-1-SQOOP_SERVER-BASE", "SQOOP_CLIE
NT-1-GATEWAY-BASE", "YARN-1-GATEWAY-BASE", "YARN-1-JOBHISTORY-BASE", "YA
RN-1-RESOURCEMANAGER-BASE", "ZOOKEEPER-1-SERVER-BASE" ]
```

```
  }, {
    "refName" : "HostTemplate-1-from-myCluster-4.myDomain.com",
    "cardinality" : 1,
    "roleConfigGroupsRefNames" : [ "FLUME-1-AGENT-BASE", "HBASE-1-REGI
ONSERVER-BASE", "HDFS-1-DATANODE-BASE", "HIVE-1-GATEWAY-BASE", "IMPALA-1
-IMPALAD-BASE", "KUDU-1-KUDU_TSERVER-BASE", "MAPREDUCE-1-TASKTRACKER-BAS
E", "SPARK_ON_YARN-1-GATEWAY-BASE", "SQOOP_CLIENT-1-GATEWAY-BASE", "YARN
-1-NODEMANAGER-BASE" ]
  }, {
    "refName" : "HostTemplate-2-from-myCluster-[2-3].myDomain.com",
    "cardinality" : 2,
    "roleConfigGroupsRefNames" : [ "FLUME-1-AGENT-BASE", "HBASE-1-REGIO
NSERVER-BASE", "HDFS-1-DATANODE-BASE", "HIVE-1-GATEWAY-BASE", "IMPALA-1-
IMPALAD-BASE", "KAFKA-1-KAFKA_BROKER-BASE", "KUDU-1-KUDU_TSERVER-BASE",
"MAPREDUCE-1-TASKTRACKER-BASE", "SPARK_ON_YARN-1-GATEWAY-BASE", "SQOOP_C
LIENT-1-GATEWAY-BASE", "YARN-1-NODEMANAGER-BASE" ]
  } ],
  "instantiator" : {
    "clusterName" : "myCluster_new",
    "hosts" : [ {
      "hostName" : "myNewCluster-1.myDomain.com",
      "hostTemplateRefName" : "HostTemplate-0-from-myCluster-1.myDomain.
com",
      "roleRefNames" : [ "HDFS-1-NAMENODE-c975a0b51fd36e914896cd5e0adb
1b5b" ]
    }, {
      "hostName" : "myNewCluster-5.myDomain.com",
      "hostTemplateRefName" : "HostTemplate-1-from-myCluster-4.myDoma
in.com"
    }, {
      "hostNameRange" : "myNewCluster-[3-4].myDomain.com",
      "hostTemplateRefName" : "HostTemplate-2-from-myCluster-[2-3].myDom
ain.com"
    } ],
```

2. For host sections that have a roleRefNames line, determine the role type and assign the appropriate host for the role. If there are multiple instances of a role, you must select the correct hosts. To determine the role type, search the template file for the value of roleRefNames.

   For example: For a role ref named HDFS-1-NAMENODE-0be88b55f5dedbf7bc74d61a86c0253e, if you search for that string, you find a section similar to the following:

   ```
   "roles": [
   {
   "refName": "HDFS-1-NAMENODE-0be88b55f5dedbf7bc74d61a86c0253e",
   "roleType": "NAMENODE"
   }
   ]
   ```

   In this case, the role type is NAMENODE.

3. Modify the variables section. This section contains various properties from the source cluster. You can change any of these values to be different in the new cluster, or you can leave the values as copied from the source. For any values shown as <changeme>, you must provide the correct value.

   Note: Many of these variables contain information about databases used by the Hive Metastore and other Runtime components. Change the values of these variables to match the databases configured for the new cluster.

4. Enter the internal name of the new cluster on the line with "clusterName" : "<changeme>". For example:

   ```
   "clusterName" : "QE_test_cluster"
   ```

5. (Optional) Change the display name for the cluster. Edit the line that begins with "displayName" (near the top of the JSON file); for example:

```
"displayName" : "myNewCluster",
```

Importing the Template to a New Cluster

Complete the steps below to import the cluster template.

Before you begin

You can only import a cluster template to create a new cluster that is running the same version of Cloudera Runtime as the cluster from which the template was exported.

Procedure

1. (Optional) If you are restoring the configuration from a redacted backup JSON file, you must replace all entries with the word "REDACTED" with the appropriate value.
2. Log in to the Cloudera Manager server as root.
3. Run the following command to import the template. If you have remote repository URLS configured in the source cluster, append the command with ?addRepositories=true.

```
curl -X POST -H "Content-Type: application/json" -d
      @path_to_template/template_filename.json
ht
tp://admin_user:admin_password@cloudera_manager_url:cloudera_manager_port/
api/v43/cm/importClusterTemplate
```

You should see a response similar to the following:

```
{
  "id" : 17,
  "name" : "ClusterTemplateImport",
  "startTime" : "2020-03-09T23:44:38.491Z",
  "active" : true,
  "children" : {
    "items" : [ ]
  }
```

Examples:

```
curl -X POST -H "Content-Type: application/json" -d @myTemplate.json ht
tp://admin:admin@myNewCluster-1.mydomain.com:7182/api/v43/cm/importCluste
rTemplate
```

```
curl -X POST -H "Content-Type: application/json" -d @myTemplate.json ht
tp://admin:admin@myNewCluster-1.mydomain.com:7182/api/v43/cm/importCluste
rTemplate?addRepositories=true
```

If there is no response, or you receive an error message, the JSON file may be malformed, or the template may have invalid hostnames or invalid references. Inspect the JSON file, correct any errors, and then re-run the command.

4. Open Cloudera Manager for the new cluster in a web browser and click the Cloudera Manager logo to go to the home page.

5. Click the All Recent Commands tab.

   If the import is proceeding, you should see a link labeled Import Cluster Template. Click the link to view the progress of the import.

   If any of the commands fail, correct the problem and click Retry. You may need to edit some properties in Cloudera Manager.

   After you import the template, Cloudera Manager applies the Autoconfiguration rules that set properties such as memory and CPU allocations for various roles. If the new cluster has different hardware or operational requirements, you may need to modify these values.

Sample Python Code

You can perform the steps to export and import a cluster template programmatically using a client written in Python or other languages. (You can also use the curl commands provided above.)

Python export example:

```
resource = ApiResource("myCluster-1.myDomain.com", 7180, "admin", "admin", v
ersion=43)
cluster = resource.get_cluster("Cluster1");
template = cluster.export(False)
pprint(template)
```

Python import example:

```
resource = ApiResource("localhost", 8180, "admin", "admin", version=43)
 with open('~/cluster-template.json') as data_file:
   data = json.load(data_file)
template = ApiClusterTemplate(resource).from_json_dict(data, resource)
cms = ClouderaManager(resource)
cms.import_cluster_template(template)
```