

Cloudera Director User Guide



Important Notice

© 2010-2016 Cloudera, Inc. All rights reserved.

Cloudera, the Cloudera logo, Cloudera Impala, and any other product or service names or slogans contained in this document are trademarks of Cloudera and its suppliers or licensors, and may not be copied, imitated or used, in whole or in part, without the prior written permission of Cloudera or the applicable trademark holder.

Hadoop and the Hadoop elephant logo are trademarks of the Apache Software Foundation. All other trademarks, registered trademarks, product names and company names or logos mentioned in this document are the property of their respective owners. Reference to any products, services, processes or other information, by trade name, trademark, manufacturer, supplier or otherwise does not constitute or imply endorsement, sponsorship or recommendation thereof by us.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Cloudera.

Cloudera may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Cloudera, the furnishing of this document does not give you any license to these patents, trademarks copyrights, or other intellectual property. For information about patents covering Cloudera products, see <http://tiny.cloudera.com/patents>.

The information in this document is subject to change without notice. Cloudera shall not be liable for any damages resulting from technical errors or omissions which may be present in this document, or from use of this document.

Cloudera, Inc.

1001 Page Mill Road, Bldg 3

Palo Alto, CA 94304

info@cloudera.com

US: 1-888-789-1488

Intl: 1-650-362-0488

www.cloudera.com

Release Information

Version: Cloudera Director 1.5.x

Date: June 21, 2016

Table of Contents

Introduction.....	7
Cloudera Director Features.....	7
Cloudera Director Client and Server.....	8
<i>Cloudera Director Client.....</i>	<i>8</i>
<i>Cloudera Director Server.....</i>	<i>9</i>
Displaying Cloudera Director Documentation.....	10
Cloudera Director Release Notes.....	11
New Features and Changes in Cloudera Director.....	11
<i>New Features and Changes in Cloudera Director 1.....</i>	<i>11</i>
Known Issues and Workarounds in Cloudera Director 1.....	12
<i>Default Memory Autoconfiguration for Monitoring Services May Be Suboptimal.....</i>	<i>12</i>
<i>Changes to Cloudera Manager Username and Password Must Also Be Made in Cloudera Director.....</i>	<i>12</i>
<i>Cloning and Growing a Kerberos-Enabled Cluster Fails.....</i>	<i>12</i>
<i>Cloudera Director Does Not Sync With Cluster Changes Made in Cloudera Manager.....</i>	<i>12</i>
<i>Kafka With a Cloudera Manager Version of 5.4.x and Lower Causes Failure.....</i>	<i>12</i>
<i>Cloudera Director May Use AWS Credentials From Instance of Cloudera Director Server.....</i>	<i>13</i>
<i>Root Partition Resize Fails on CentOS 6.5 (HVM).....</i>	<i>13</i>
<i>Cloudera Director Does Not Set Up External Databases for Oozie, Hue, and Sqoop2.....</i>	<i>13</i>
<i>Terminating Clusters That are Bootstrapping Must be Terminated Twice for the Instances to be Terminated.....</i>	<i>13</i>
<i>When Using RDS and MySQL, Hive Metastore Canary May Fail in Cloudera Manager.....</i>	<i>13</i>
Fixed Issues.....	13
<i>Issues Fixed in Cloudera Director 1.5.2.....</i>	<i>13</i>
<i>Issues Fixed in Cloudera Director 1.5.1.....</i>	<i>14</i>
<i>Issues Fixed in Cloudera Director 1.5.0.....</i>	<i>14</i>
<i>Issues Fixed in Cloudera Director 1.1.3.....</i>	<i>14</i>
<i>Issues Fixed in Cloudera Director 1.1.2.....</i>	<i>15</i>
<i>Issues Fixed in Cloudera Director 1.1.1.....</i>	<i>15</i>
Requirements and Supported Versions.....	16
Cloud Providers.....	16
Cloudera Director Service Provider Interface (SPI).....	16
Supported Software and Distributions.....	16
Resource Requirements.....	17
Supported Cloudera Manager and CDH Versions.....	17
Networking and Security Requirements.....	17

Supported Browsers.....	18
Getting Started with Cloudera Director.....	19
Connecting to Your Cluster Using a SOCKS Proxy.....	19
<i>Step 1: Create a Proxy Auto-Config File.....</i>	<i>19</i>
<i>Step 2: Set Up SwitchySharp.....</i>	<i>19</i>
<i>Step 3: Set Up a SOCKS Proxy with SSH.....</i>	<i>20</i>
Getting Started on Amazon Web Services (AWS).....	20
<i>Setting up the AWS Environment.....</i>	<i>20</i>
<i>Choosing an AMI.....</i>	<i>22</i>
<i>Creating an EC2 Instance for Cloudera Director.....</i>	<i>23</i>
<i>Installing Cloudera Director Server on the EC2 Instance.....</i>	<i>26</i>
<i>Deploying Cloudera Manager and CDH on AWS.....</i>	<i>28</i>
Cloudera Director Client.....	33
Installing Cloudera Director Client.....	33
Provisioning a Cluster on AWS.....	34
Running Cloudera Director Client.....	35
Using the Command Line Interface.....	37
<i>Local commands.....</i>	<i>37</i>
<i>Remote commands.....</i>	<i>37</i>
Connecting to Cloudera Manager.....	38
Modifying a Cluster with Cloudera Director Client.....	39
<i>Growing or Shrinking a Cluster.....</i>	<i>39</i>
Managing Cloudera Manager Instances with Cloudera Director Server.....	40
Submitting a Cluster Configuration File.....	40
Deploying Clusters in an Existing Environment.....	40
Cloudera Manager Health Information.....	41
Opening Cloudera Manager.....	41
Adding HDFS DataNodes to a Cluster.....	42
Removing or Repairing Hosts in a Cluster.....	42
<i>Removing Hosts from a Cluster.....</i>	<i>42</i>
<i>Repairing Hosts in a Cluster.....</i>	<i>43</i>
Terminating a Cluster.....	43
<i>Terminating a Cluster with the UI.....</i>	<i>43</i>
<i>Terminating a Cluster with the CLI.....</i>	<i>43</i>
Starting and Stopping the Cloudera Director Server.....	44
User Management.....	44
<i>Managing Users with the Cloudera Director Web UI.....</i>	<i>44</i>
<i>Managing Users with the Cloudera Director API.....</i>	<i>45</i>

Customization and Advanced Configuration.....47

The Cloudera Director Configuration File.....	47
<i>Location of Sample Configuration Files.....</i>	<i>47</i>
<i>Customizing the Configuration File.....</i>	<i>47</i>
Creating a Cloudera Manager and CDH AMI.....	47
Choosing an AMI.....	47
<i>Finding Available AMIs.....</i>	<i>48</i>
Creating AWS Identity and Access Management (IAM) Policies.....	48
Using MySQL for Cloudera Director Server.....	50
<i>Installing the MySQL Server.....</i>	<i>50</i>
<i>Configuring and Starting the MySQL Server.....</i>	<i>51</i>
<i>Installing the MySQL JDBC Driver.....</i>	<i>52</i>
<i>Creating a Database for Cloudera Director Server.....</i>	<i>53</i>
<i>Configuring Cloudera Director Server to use the MySQL Database.....</i>	<i>54</i>
Cloudera Director Database Encryption.....	54
<i>Cipher Configuration.....</i>	<i>54</i>
<i>Starting with Encryption.....</i>	<i>55</i>
<i>Changing Encryption.....</i>	<i>56</i>
Using an External Database for Cloudera Manager and Clusters.....	57
<i>Defining Database Servers.....</i>	<i>57</i>
<i>Using External Databases.....</i>	<i>61</i>
Setting Cloudera Director Properties.....	64
Setting Cloudera Manager Configurations.....	72
<i>Setting up a Cloudera Manager License.....</i>	<i>73</i>
<i>Deployment Template Configuration.....</i>	<i>73</i>
<i>Cluster Template Service-wide Configuration.....</i>	<i>74</i>
<i>Cluster Template Roletype Configurations.....</i>	<i>75</i>
Configuring Cloudera Director for a New AWS Instance Type.....	76
<i>Updated Virtualization Mappings.....</i>	<i>76</i>
<i>Updated Ephemeral Device Mappings.....</i>	<i>76</i>
<i>Using the New Mappings.....</i>	<i>77</i>
Post-creation Scripts.....	77
<i>Configuring the Scripts.....</i>	<i>77</i>
<i>Predefined Environment Variables</i>	<i>77</i>
Enabling Sentry Service Authorization.....	78
<i>Prerequisites.....</i>	<i>78</i>
<i>Setting Up the Sentry Service Using the Cloudera Director CLI.....</i>	<i>78</i>
<i>Setting up the Sentry Service Using the Cloudera Director API.....</i>	<i>79</i>
<i>Related Links.....</i>	<i>80</i>

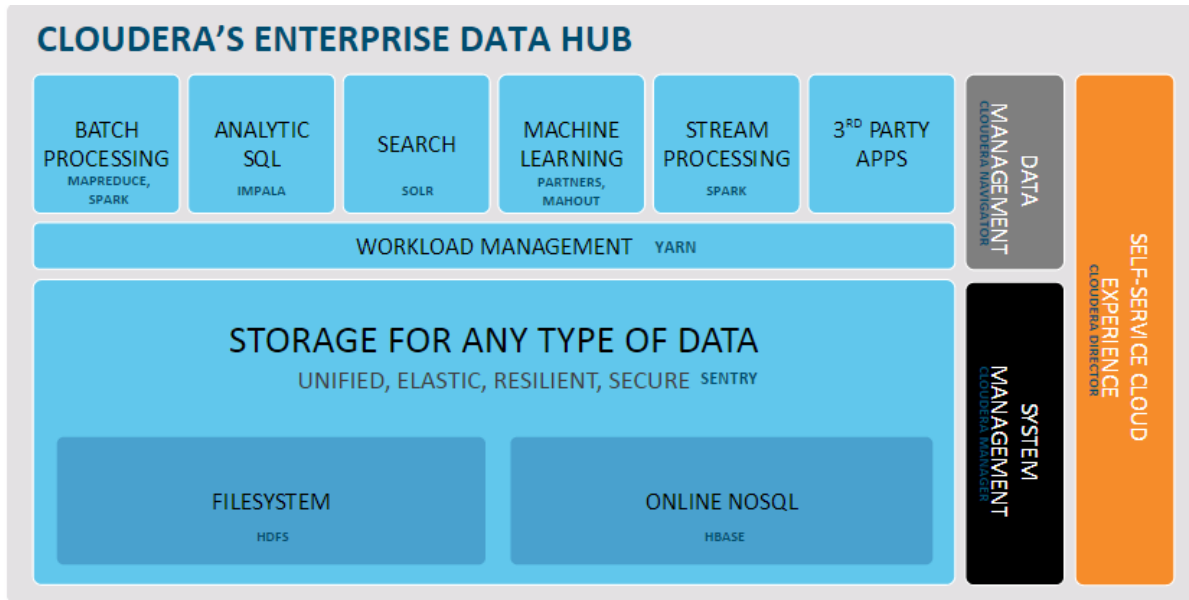
Upgrading Cloudera Director.....81

Before Upgrading Cloudera Director from 1.1.x to 1.5.x.....	81
Upgrading Cloudera Director 1.1.x to Cloudera Director 1.5.x.....	81
Troubleshooting Cloudera Director.....	84
DNS Issues.....	84
Server doesn't start.....	85
Problem When Removing Hosts from a Cluster.....	85
Problems Connecting to Cloudera Director Server.....	86
Frequently Asked Questions.....	87
General Questions.....	87
Cloudera Director Glossary.....	88

Introduction

Cloudera Director enables reliable self-service for CDH and Cloudera Enterprise Data Hub in the cloud.

It is designed to provide a single-pane-of-glass administration experience for central IT to reduce costs and deliver agility, and for end-users to easily provision and scale clusters. Advanced users can interact with Cloudera Director programmatically through the REST API or the CLI to maximize time-to-value for an enterprise data hub in cloud environments.



Cloudera Director is designed for both long running and ephemeral clusters. With long running clusters, you deploy one or more clusters that you can scale up or down to adjust to demand. With ephemeral clusters, you can launch a cluster, schedule any jobs, and shut the cluster down after the jobs complete.

Running Cloudera in the cloud supports:

- Faster procurement—Deploying servers in the cloud is faster than completing a lengthy hardware acquisition process.
- Easier scaling—To meet changes in cluster demand, it is easier to add and remove new hosts in the cloud than in a bare metal environment.
- Infrastructure migration—Many organizations have already moved to a cloud architecture, while others are in the process of moving.

Cloudera Director Features

Cloudera Director provides a rich set of features for launching and managing clusters in cloud environments. The following table describes the benefits of using Cloudera Director.

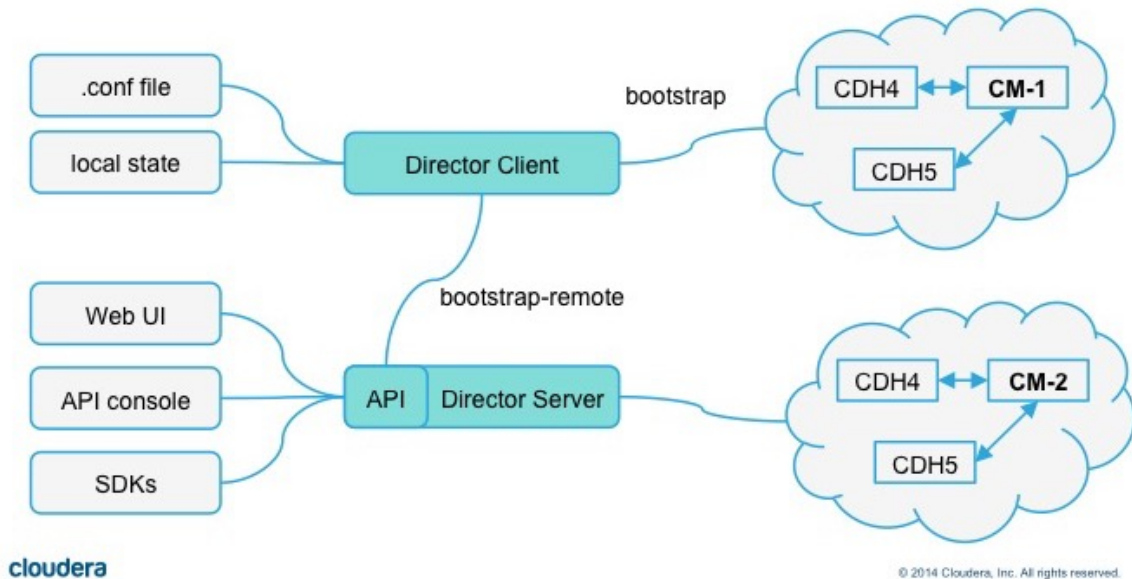
Benefit	Features
Simplified cluster life cycle management	Simple user interface: <ul style="list-style-type: none"> • Self-Service spin up and tear down • Dynamic scaling for spiky workloads • Simple cloning of clusters • Cloud blueprints for repeatable deployments

Benefit	Features
Elimination of lock-in	Flexible, open platform: <ul style="list-style-type: none"> • 100% open source Hadoop distribution • Native support for hybrid deployments • Third-party software deployment in the same workflow • Support for custom, workload-specific deployments
Accelerated time to value	Enterprise-ready security and administration: <ul style="list-style-type: none"> • Support for complex cluster topologies • Minimum size cluster when capacity constrained • Management tooling • Compliance-ready security and governance • Backup and disaster recovery with an optimized cloud storage connector
Reduced support costs	Monitoring and metering tools: <ul style="list-style-type: none"> • Multi-cluster health dashboard • Instance tracking for account billing

Cloudera Director Client and Server

Cloudera Director supports cluster deployment through the client or the server.

The diagram below illustrates the components of Cloudera Director. At the center of the diagram are the two main components: the Cloudera Director client and Cloudera Director server.



Cloudera Director Client

The Cloudera Director client is a standalone process, with no UI and no server. It provides the simplest way of using Cloudera Director. You interact at the command line through the host on which the client is installed. You start the client process with the `bootstrap` command, and everything runs in a single process, with all configurations specified

in the `.conf` file. When you are done, issue the `terminate` command to stop the process. If you want to run Cloudera Director repeatedly with the same settings, your `.conf` file preserves those settings and can be reused as is or with modifications.

In the diagram above, the lines that extend from the client show that the client stores its state locally. The client uses the `.conf` file to launch clusters, either directly by using the `bootstrap` command, or through the server by using the `bootstrap-remote` command, described below in [Using Cloudera Director Server](#) on page 9. For more information about the `.conf` file, see [The Cloudera Director Configuration File](#) on page 47.

Cloudera does not recommend the standalone client mode for production use, but it can be used for development work, proof-of-concept demonstrations, or trying the product.

Cloudera Director Server

The Cloudera Director server is designed for a more centralized environment, managing multiple Cloudera Manager instances and CDH clusters with multiple users and user accounts. You can log into the server UI and launch clusters, or you can send the server a cluster configuration file from the Cloudera Director client using the `bootstrap-remote` command. Use the server to launch and manage large numbers of clusters in a production environment.

In the diagram above, the lines that extend from the server show three interfaces to the server: the Web UI, the API console, and the SDKs. All three interfaces interact with the server through the API, represented in this diagram as part of the Cloudera Director server component. The line to the right indicates that the server, like the client, can launch Cloudera Manager instances and CDH clusters. The processes that interact with the cloud infrastructure run on the server, and the server owns the state for the clusters it has launched.

Using Cloudera Director Server

You can interact with Cloudera Director server in several ways. The best way for you depends on your purposes and whether you want to use the Cloudera-recommended default configurations, or if instead you require customized configurations for a particular use case or environment.

With the User Interface (UI) Only

The UI provides a view of the components present in your setup, including the clusters and processes that are running; the health of cluster components; and easy access to error logs. You can also use the UI for initial setups, and interact with Cloudera Director server through the UI only, without using the APIs or the `.conf` file.

This mode can be used in production setups, but Cloudera recommends that it be used for simple setups offered through the guided setup wizard that the UI provides, and not for customized setups. Although many advanced features are available using only the UI, it is not ideal for experimenting with configurations because your configuration settings are not preserved, making iteration difficult. For ease of iteration with customized setups, choose a mode that uses the `.conf` file, where your settings will be preserved, or the API, where your settings can be saved, for example, in a Python script.

With the Command Line Interface (CLI)

With the CLI, if the server is running, you can set up a cluster using the `bootstrap-remote` command with the `.conf` file. In this mode, the UI can be used to get information about the clusters and services that have been set up and the processes running on different clusters.

When used to dispatch the `.conf` file to a server through `bootstrap-remote`, the Cloudera Director client provides full access to all advanced features, such as custom configurations of Cloudera Manager services and hosts. If you use this mode and want to iterate with the same settings, you can use the `.conf` file as a record of the settings. You can set up new clusters later by simply using the UI mode and entering the settings preserved in this `.conf` file.

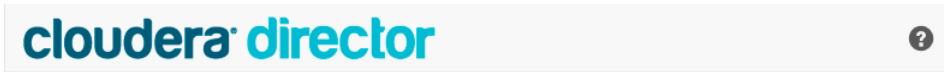
For maximum flexibility and power—for example, if you want to experiment with custom configurations and custom role assignments—using `bootstrap-remote` with the `.conf` file is a good choice.

With the API

For programmatic interaction with the server, Cloudera Director includes SDKs for Python and Java, and an API console. You can use the API to access advanced Cloudera Director features, including custom configuration settings. As with the `.conf` file, using the API supports iteration because your settings can be saved in a Python script or Java file.

Displaying Cloudera Director Documentation

To display Cloudera Director documentation for any page in the server UI, click the question mark icon in the upper-right corner at the top of the page:



The latest help files are hosted on the Cloudera web site, but help files are also embedded in the product for users who do not have Internet access. By default, the help files displayed when you click the question mark icon are those hosted on the Cloudera web site because these include the latest updates. You can configure Cloudera Director to open either the latest help from the Cloudera web site or locally installed help by toggling the value of `lp.webapp.documentationType` to `ONLINE` or `EMBEDDED` in the server `application.properties` configuration file.

Cloudera Director Release Notes

These Release Notes provide information on the new features and known issues and limitations for Cloudera Director 1.

For information about supported operating systems, and other requirements for using Cloudera Director, see [Cloudera Director Requirements and Supported Versions](#).

New Features and Changes in Cloudera Director

New Features and Changes in Cloudera Director 1

The following sections describe what's new and changed in each Cloudera Director 1 release.

What's New in Cloudera Director 1.5.2

- Cloudera Director now supports RHEL 6.7.
- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 1.5.2](#) for details.

What's New in Cloudera Director 1.5.1

- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 1.5.1](#) on page 14 for details.

What's New in Cloudera Director 1.5.0

- Cloudera Director now supports multiple cloud providers through an open-source plugin interface, the [Cloudera Director Service Provider Interface \(Cloudera Director SPI\)](#).
- Google Cloud Platform is now supported via an open-source implementation of the Cloudera Director SPI, the [Cloudera Director Google Plugin](#).
- Database servers set up by Cloudera Director can now be managed from the UI.
- Users can now specify custom scripts to be run after cluster creation. Example scripts for enabling HDFS High Availability and Kerberos are available on the [Cloudera GitHub site](#).
- The Cloudera Director database can now be encrypted. Encryption is enabled by default for new installations.
- Cluster and Cloudera Manager configurations can now be set via the UI.
- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 1.5.0](#) on page 14 for details.

What's New in Cloudera Director 1.1.3

- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 1.1.3](#) on page 14 for details.
- Cloudera Director's disk preparation method now supports RHEL 6.6, which is supported by Cloudera Manager 5.4.
- Custom endpoints for AWS Identity and Access Management (IAM) are now supported.
- To ensure version compatibility between Cloudera Manager and CDH, Cloudera Director now defaults to installing the latest 5.3 version of Cloudera Manager and CDH, rather than installing the latest post-5.3 version.

What's New in Cloudera Director 1.1.2

- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 1.1.2](#) for details.

What's New in Cloudera Director 1.1.1

- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 1.1.1](#) for details.

What's New in Cloudera Director 1.1.0

- Support for demand-based shrinking of clusters
- Integration with Amazon RDS to enable end-to-end setup of clusters as well as related databases

- Native client bindings for Cloudera Director API in Java and Python
- Faster bootstrap of Cloudera Manager and clusters
- Improved User Interface of Cloudera Director server including display of health of clusters and ability to customize cluster setups
- Improvements to usability and documentation

Known Issues and Workarounds in Cloudera Director 1

The following sections describe the current known issues in Cloudera Director 1.

Default Memory Autoconfiguration for Monitoring Services May Be Suboptimal

Depending on the size of your cluster and your choice of instance types, you may need to manually increase the memory limits for the Host Monitor and Service Monitor. Cloudera Manager displays a configuration validation warning or error if the memory limits are insufficient.

Workaround: Override `firehose_heapsize` for `HOSTMONITOR` and `SERVICES` with a different value in bytes (for example, 536900000 for ~512 MB). Cloudera also recommends using instances with a minimum of 15 GB of memory for management roles (30 GB recommended).

Changes to Cloudera Manager Username and Password Must Also Be Made in Cloudera Director

If the Cloudera Manager username and password are changed directly in Cloudera Manager, Cloudera Director can no longer add new instances or authenticate with Cloudera Manager. Username and password changes must be implemented in Cloudera Director as well.

Workaround: Use the [update-deployment.py](#) script to update the Cloudera Manager credentials in Cloudera Director:

```
$ wget
https://raw.githubusercontent.com/cloudera/director-scripts/master/util/update-deployment.py

$ sudo pip install cloudera-director-python-client
$ python update-deployment.py --admin-username admin --admin-password admin
--server "http://<director_server_host>:7189" --environment <environment_name>
--deployment <deployment_name> --deployment-password newPassword
```

In the example, the deployment username, `admin`, is not changed. The password is changed to `newPassword`.

Cloning and Growing a Kerberos-Enabled Cluster Fails

Cloning of a cluster that is using Kerberos authentication fails, whether it is cloned manually or by using the `kerberize-cluster.py` script. Growing a cluster that is using Kerberos authentication fails.

Workaround: None.

Cloudera Director Does Not Sync With Cluster Changes Made in Cloudera Manager

Modifying a cluster in Cloudera Manager after it is bootstrapped does not cause the cluster's state to be synchronized with Cloudera Director. Services that have been added or removed in Cloudera Manager do not show up in Cloudera Director when growing the cluster.

Workaround: None.

Kafka With a Cloudera Manager Version of 5.4.x and Lower Causes Failure

Kafka installed with a Cloudera Manager version of 5.4.x and lower causes the Cloudera Manager first run wizard, and therefore the bootstrap process, to fail, unless you override the configuration setting `broker_max_heap_size`.

Workaround: Override `broker_max_heap_size` by setting it to at least 256 MB.

Cloudera Director May Use AWS Credentials From Instance of Cloudera Director Server

Cloudera Director server uses the AWS credentials from a configured Environment, as defined in a client configuration file or through the Cloudera Director UI. If the Environment is not configured with credentials in Cloudera Director, the Cloudera Director server instead uses the AWS credentials that are configured on the instance on which the Cloudera Director server is running. When those credentials differ from the intended ones, EC2 instances may be allocated under unexpected accounts. Ensure that the Cloudera Director server instance is not configured with AWS credentials.

Severity: Medium

Workaround: Ensure that the Cloudera Director Environment has correct values for the keys. Alternatively, use IAM profiles for the Cloudera Director server instance.

Root Partition Resize Fails on CentOS 6.5 (HVM)

Cloudera Director cannot resize the root partition on Centos 6.5 HVM AMIs. This is caused by a bug in the AMIs. For more information, see the [CentOS Bug Tracker](#).

Workaround: None.

Cloudera Director Does Not Set Up External Databases for Oozie, Hue, and Sqoop2

Cloudera Director cannot set up external databases for Oozie, Hue, and Sqoop2.

Workaround: Set up the databases for these services as described in [Cloudera Manager and Managed Service Databases](#). Provide the database properties such as `host` address and `username` to Cloudera Director in the relevant Oozie service configuration section.

Terminating Clusters That are Bootstrapping Must be Terminated Twice for the Instances to be Terminated

Terminating a cluster that is bootstrapping stops ongoing processes but keeps the cluster in the bootstrapping phase.

Severity: Low

Workaround: To transition the cluster to the **Terminated** phase, terminate the cluster again.

When Using RDS and MySQL, Hive Metastore Canary May Fail in Cloudera Manager

If you include Hive in your clusters and configure the Hive metastore to be installed on MySQL, Cloudera Manager may report, "The Hive Metastore canary failed to create a database." This is caused by a MySQL bug in MySQL 5.6.5 or later that is exposed when used with the MySQL JDBC driver (used by Cloudera Director) version 5.1.19 or earlier. For information on the MySQL bug, see the [MySQL bug description](#).

Workaround: Depending on the driver version installed by Cloudera Director from your platform's software repositories, select an older MySQL version that does not have this bug.

Fixed Issues

The following sections describe fixed issues in each Cloudera Director 1 release.

Issues Fixed in Cloudera Director 1.5.2

Apache Commons Collections Deserialization Vulnerability

Cloudera has learned of a potential security vulnerability in a third-party library called the [Apache Commons Collections](#). This library is used in products distributed and supported by Cloudera ("Cloudera Products"), including Cloudera Director. At this time, no specific attack vector for this vulnerability has been identified as present in Cloudera Products.

The Apache Commons Collections potential security vulnerability is titled "Arbitrary remote code execution with InvokerTransformer" and is tracked by [COLLECTIONS-580](#). MITRE has not issued a CVE, but related [CVE-2015-4852](#) has been filed for the vulnerability. CERT has issued [Vulnerability Note #576313](#) for this issue.

Releases affected: Cloudera Director 1.5.1 and lower, CDH 5.5.0, CDH 5.4.8 and lower, Cloudera Manager 5.5.0, Cloudera Manager 5.4.8 and lower, Cloudera Navigator 2.4.0, and Cloudera Navigator 2.3.8 and lower

Users affected: All

Severity (Low/Medium/High): High

Impact: This potential vulnerability may enable an attacker to execute arbitrary code from a remote machine without requiring authentication.

Immediate action required: Upgrade to Cloudera Director 1.5.2, Cloudera Manager 5.5.1, and CDH 5.5.1.

Serialization for Complex Nested Types in Python API Client

Serialization for complex nested types has been fixed in the Python API client.

Issues Fixed in Cloudera Director 1.5.1

Support for Configuration Keys Containing Special Characters

Configuration file parsing has been updated to correctly support quoted configuration keys containing special characters such as colons and periods. This enables the usage of special characters in service and role type configurations, and in instance tag keys.

Issues Fixed in Cloudera Director 1.5.0

Growing clusters may fail when using a repository URL that only specifies major and minor versions

When using a Cloudera Manager package repository or CDH/parcel repository URL that only specifies the major or minor versions, Cloudera Director may incorrectly use the latest available version when trying to grow a cluster.

Workaround: Use a parcel repository that is specified down to the three-digit maintenance version. This will ensure that Director finds the correct version during cluster growth.

For Cloudera Manager: http://archive.cloudera.com/cm5/redhat/6/x86_64/cm/5.3.3/

For CDH: <http://archive.cloudera.com/cdh5/parcels/5.3.3/>

Flume doesn't start automatically after FirstRun

Although you can deploy Flume through Cloudera Director, you must start it manually using Cloudera Manager after Cloudera Director bootstraps the cluster.

Workaround: Start Flume manually using Cloudera Manager after Cloudera Director bootstraps the cluster.

Impala daemons attempt to connect over IPv6

Impala daemons attempt to connect over IPv6.

Workaround: Add the following command part of the instance bootstrap script: `sysctl -w net.ipv6.conf.all.disable_ipv6=1`.

DNS queries occasionally timeout with AWS VPN

DNS queries occasionally timeout with AWS VPN.

Workaround: Cloudera recommends that you install NSCD (name service cache daemon) on all cluster instances via a bootstrap script. By default Linux does not cache DNS lookups. For more information, see the [Linux NSCD man page](#).

Issues Fixed in Cloudera Director 1.1.3

Ensure accurate time on startup

Instance normalization has been improved to ensure time is synchronized by Network Time Protocol (NTP) prior to bootstrapping, which improves cluster reliability and consistency.

Speed up ephemeral drive preparation

Instance drive preparation during the bootstrapping process was slow, especially for instances with many large ephemeral drives. Time required for this process has been reduced.

Fix typographical error in the `virtualizationmappings.properties` file

The `d2` instance type `d2.4xlarge` was incorrectly entered into Cloudera Director as `d3.4xlarge` in `virtualizationmappings.properties`. This has been corrected.

Avoid upgrading pre-installed Cloudera Manager packages

Cloudera Director no longer upgrades pre-installed Cloudera Manager packages.

Issues Fixed in Cloudera Director 1.1.2

Parcel validation fails when using HTTP proxy

Parcel validation now works when configuring an HTTP proxy for Cloudera Director server, allowing correctly configured parcel repository URLs to be used as expected.

Unable to grow a cluster after upgrading Cloudera Director 1.0 to 1.1.0 or 1.1.1

Cloudera Director now sets up parcel repository URLs correctly when a cluster is modified.

Add support for `d2` and `c4` AWS instance types

Cloudera Director now includes first-class support for new AWS instance types `d2` and `c4`. Cloudera Director can be configured to use additional instance types at any point as they become available in AWS.

Issues Fixed in Cloudera Director 1.1.1

Service level custom configurations are ignored

Restored the ability to have service level custom configurations. Due to internal refactoring changes, it was no longer possible to override service level configs.

The property `customBannerText` is ignored and not handled as a deprecated property

Restored the `customBannerText` configuration file property, which was removed during the internal refactoring work.

Fixed progress bar issues when a job fails

The UI showed a progress bar even when a job had failed.

Updated IAM Help text on Add Environment page

The help text on the Add Environment page for Role-based keys should refer to AWS Identity and Access Management (IAM), not to AMI.

Add `eu-central-1` to the region dropdown

The `eu-central-1` region has been added to the region dropdown on the Add Environment page.

Gateway roles should assign YARN, HDFS, and Spark gateway roles

All available gateway roles, including YARN, HDFS, and Spark, should be deployed by default on the instance.

Spark on YARN should be shown on the Modify Cluster page

Spark on YARN did not appear in the list of services on the Modify Cluster page.

Requirements and Supported Versions

The following sections describe the requirements and supported operating systems, databases, and browsers for Cloudera Director.

Cloud Providers

Cloudera Director has native support for Amazon Web Services (AWS) and Google Cloud Platform.

Each Cloudera Director release embeds the current plugin for supported cloud providers, but a newer plugin may have been posted on the Cloudera GitHub site subsequent to the Cloudera Director release. To check for the latest version, click the appropriate link:



- [AWS cloud provider plugin](#)
- [Google Cloud Platform cloud provider plugin](#)

Cloudera Director Service Provider Interface (SPI)

The Cloudera Director SPI defines an open source Java interface that plugins implement to add support for additional cloud providers to Cloudera Director. For more information, see the README.md file for the SPI [Cloudera Director GitHub repository](#).

Supported Software and Distributions

The table below lists software requirements, recommendations, and supported versions for resources used with Cloudera Director.

	Cloudera Director	Cloudera Manager and CDH
Operating Systems (64-bit only)	RHEL and CentOS 5.7, 6.4, 6.5, and 6.7 SLES Server 11 Wheezy 7.0 and 7.1 Ubuntu 12.04 and 14.04	RHEL and CentOS 6.4, 6.5, 6.6, and 6.7  Note: RHEL 6.6 supports AWS D2 instances. The other RHEL and CentOS versions do not support these instance types.
Oracle Java SE Development Kit (JDK)	Oracle JDK version 6, 7, or 8  Note: For download and installation information, see Java SE Downloads .	Oracle JDK version 6 or 7 (Cloudera Manager 4 and CDH 4) Oracle JDK version 7 or 8 (Cloudera Manager 5 and CDH 5)
Default Database	Embedded H2 database	Embedded PostgreSQL Database
Supported Databases	MySQL 5.5, 5.6	MySQL 5.5, 5.6 PostgreSQL 8.1, 8.3, 8.4, and 9.1



Note: By default, Cloudera Director stores its environment and cluster data in an embedded H2 database located at `/var/lib/cloudera-director-server/state.h2.db`. Back up this file to avoid losing the data. For information on using an external MySQL database in place of the H2 embedded database, see [Using MySQL for Cloudera Director Server](#) on page 50. Cloudera recommends using an external database for both Cloudera Director and Cloudera Manager for production environments.

Resource Requirements

The table below lists requirements for resources used with Cloudera Director.

	Cloudera Director	Cloudera Manager and CDH
CPU	2	4
RAM	3.75 GB	64 GB
Disk	8 GB	500 GB
Recommended AWS instance	c3.large and c4.large	Cloudera Manager: m4.large or m4.xlarge <div data-bbox="1088 861 1136 913" data-label="Image"></div> Note: The recommended instance for Cloudera Manager is dependent on the workload. Contact your Cloudera account representative for more information.
Recommended Google Cloud Platform instance	n1-standard-2	n1-highmem-8

Supported Cloudera Manager and CDH Versions

Cloudera Director 1.x can install any version of Cloudera Manager 5 with any CDH 4 or CDH 5 parcels. Use of CDH packages is not supported.

Networking and Security Requirements

Cloudera Director requires the following inbound ports to be open:

- **TCP ports 22:** These ports allow SSH to Cloudera Director instance.
- **All traffic across all ports within the security group:** This rule allows connectivity with all the components within the Hadoop cluster. This rule avoids numerous individual ports to be opened in the security group.

Type	Protocol	Port Range	Source
SSH (22)	TCP (6)	22	0.0.0.0/0
ALL Traffic	ALL	ALL	<i>security_group_id</i> See note paragraph below.

Requirements and Supported Versions



Note: This second rule requires the security group id. If users are creating a security group from scratch, create the security group with the SSH rule and then go back and edit the security group to add allow all traffic within the security group to be allowed.

In order to connect to the AWS network Cloudera recommends to open only these ports and set up a SOCKS proxy. Unless your network has direct connection to AWS you will have to set this up to access the Cloudera Director instance. This will be done in a later step.

Supported Browsers

Cloudera Director supports the following browsers:

- Mozilla Firefox 11 and higher
- Google Chrome
- Internet Explorer 9 and higher
- Safari 5 and higher

Getting Started with Cloudera Director

This section explains how to get Cloudera Director up and running on Amazon Web Services (AWS).

Connecting to Your Cluster Using a SOCKS Proxy

For security purposes, we recommend that you connect to your cluster using a [SOCKS proxy](#). A SOCKS proxy allows a client (your computer, for example) to connect directly and securely to a server (the Director instance).

To set up a SOCKS proxy, follow the steps below.

Step 1: Create a Proxy Auto-Config File

To create a [proxy auto-config](#) (PAC) file, perform the following tasks:

1. Open a text editor and enter the following text:

```
function regExpMatch(url, pattern) {
  try { return new RegExp(pattern).test(url); } catch(ex) { return false; }
}

function FindProxyForURL(url, host) {
  // Important: replace 172.31 below with the proper prefix for your VPC subnet
  if (shExpMatch(url, "*172.31.*")) return "SOCKS5 localhost:8157";
  if (shExpMatch(url, "*ec2*.amazonaws.com*")) return 'SOCKS5 localhost:8157';
  if (shExpMatch(url, "*.compute.internal*" || shExpMatch(url,
  "*/compute.internal*")) return 'SOCKS5 localhost:8157';
  if (shExpMatch(url, "*ec2.internal*")) return 'SOCKS5 localhost:8157';
  return 'DIRECT';
}
```

2. Save the file.

The PAC file contains the three rules needed for Cloudera Director.

Step 2: Set Up SwitchySharp

1. Open Chrome and go to [Chrome Apps](#).
2. Search for **Proxy SwitchySharp** and add to it Chrome.
3. In the **SwitchySharp Options** screen, click the **Proxy Profiles** tab and do the following:
 - a. In the **Profile Name** field, enter `AWS-Cloudera`.
 - b. Click **Automatic Configuration**.
 - c. Click **Import PAC File** and import your PAC file.
 - d. Click **Save**.
4. Click the **General** tab and do the following:
 - a. Click **Quick Switch**.
 - b. Drag **[Direct Connection]** and **AWS-Cloudera** to the **Cycled Profiles** area.
 - c. Set **Startup Profile** to **[Direct Connection]**.
 - d. Click **Save**.

Step 3: Set Up a SOCKS Proxy with SSH

- Set up a SOCKS proxy to access the EC2 instance running Cloudera Director. For example, in RHEL run the following command (with your instance information):

```
ssh -i <key-file.pem> -CND 8157 ec2-user@instance_running_director_server
```

where

- C sets up compression
- N suppresses any command execution once established
- D 8157 sets up the SOCKS 5 proxy on the port



Important: If you are using a PAC file, you must use port 8157.

Getting Started on Amazon Web Services (AWS)

Before you can install and use Cloudera Director on AWS, you have to create an environment in Amazon Virtual Private Cloud (Amazon VPC), start an instance in AWS to run Cloudera Director, and create a secure connection. This section details steps for each of these tasks.

Setting up the AWS Environment

Cloudera Director requires you to set up a VPC and create an SSH key pair in the AWS environment prior to deploying Cloudera Director. This section details the steps for each of these tasks.

To begin, log in to the [AWS Management Console](#) and make sure you are in the desired region. The current region is displayed in the upper right corner of the AWS Management Console. Click the region name to change your region.

Setting Up a VPC

Cloudera Director requires an Amazon Virtual Private Cloud (Amazon VPC) to implement its virtual environment. The AWS VPC must be set up for forward and reverse hostname resolution.

Draft Comment:

The following content describes automated setup, not supported at the moment.: <p>This section describes how to manually set up a VPC. To automate the process, you can use AWS CloudFormation with a network template. For more information, see the <xref href="http://aws.amazon.com/documentation/cloudformation/" format="html" scope="external">CloudFormation Documentation</xref> and the Cloudera Director template located at <codeph><director files>/samples/cfn-cloudera-public-subnet.template.</codeph></p>

To set up a new VPC, follow the steps below. Skip these steps if you have an existing VPC you want to use.

1. Log in to the [AWS Management Console](#) and make sure you are in the desired region. The current region is displayed in the upper right corner of the AWS Management Console. Click the region name to change your region.
2. In the AWS Management Console, select **VPC** in the Networking section.
3. Click **Start VPC Wizard**. (Click VPC Dashboard in the left side pane if the **Start VPC Wizard** button is not displayed.)
4. Select the desired VPC configuration. The easiest way to get started is to select **VPC with a Single Public Subnet**.
5. Fill out the necessary sections in the VPC wizard and then click Create **Create VPC**.

Creating Your Subnet(s)

1. In the left pane, click **Subnets**.
2. Click **Create Subnet**.
3. Fill out the details for the subnet and associate it with the VPC. Click **Yes, Create**.

Configuring your Security Group

Cloudera Director requires the following inbound ports to be open:

Type	Protocol	Port Range	Source
ALL Traffic	ALL	ALL	<i>security_group_id</i>
SSH (22)	TCP (6)	22	0.0.0.0/0



Note: By default, Cloudera Director requires unrestricted outbound connectivity. If necessary, it can be configured to use proxy servers or a local mirror of all the relevant repositories.

Creating a New Security Group:

If you need to create a new security group for Cloudera Director from scratch, follow these steps:

1. In the left pane, click **Security Groups**.
2. Click **Create Security Group**.
3. Enter a name and description. Make sure to select the VPC you created from the VPC list box.
4. Click **Yes, Create**.

Select the newly-created security group and add inbound rules as detailed in the table above.

The configured security group should look similar to the following, but with your own values in the Source column.

Type	Protocol	Port Range	Source
All traffic	All	All	sg-3e48cf58 (test-doc)
SSH	TCP	22	0.0.0.0/0

The **Custom TCP Rule** shown above facilitates access to the Cloudera Director UI through port 7189. IP tables need to be disabled when opening the port by issuing this command:

```
$ sudo service iptables off
```

For more information about security groups in AWS, see [Security Groups for Your VPC](#).

Creating an SSH Key Pair

To interact with the cluster launcher and other instances, you must create an SSH key pair or use an existing EC2 key pair. If you do not have a key pair, follow these steps:



Note: For information on importing an existing key pair, see [Amazon EC2 Key Pairs](#) in the AWS documentation.

1. Select **EC2** from the **Services** navigation list box.
2. In the left pane, click **Key Pairs**.
3. Click **Create Key Pair**. In the Create Key Pair dialog box, enter a name for the key pair and click **Create**.
4. Note the key pair name. Move the automatically downloaded private key file (with .pem extension) to a secure location and note the location.

Choosing an AMI

Cloudera Director, Cloudera Manager, and CDH support only 64-bit Linux based AMIs. For CDH and Cloudera Manager on Amazon EC2, Cloudera Director only supports RHEL and CentOS distributions. Refer to [Requirements and Supported Versions](#) for details.

Cloudera recommends doing the following when choosing an AMI:

- Search on the Community AMIs page of the AWS Management Console, as described below, for the best selection.
- HVM virtualization instead of PV virtualization. For more information on HVM and PV virtualization, see [Linux AMI Virtualization Types](#) in the AWS documentation.
- Use the most recent GA release of a specific distribution. Typically, there are multiple releases associated with a specific distribution. Cloudera recommends that you select the highest version.

Finding Available AMIs

There are two ways of finding available AMIs: with the AWS Management Console (UI) or with the command line tool `awscli`. These two ways are described in the following sections.

Using the Amazon AWS Management Console

1. Log into your AWS account and ensure you are in the desired region.
2. Click on EC2 and click the **Launch Instance** button.
3. The Choose AMI page will open with **Step 1: Choose an Amazon Machine Image**. In the left pane, you can select from the following:
 - Quick Start
 - My AMIs
 - AWS Marketplace
 - Community AMIs

Select **Community AMIs** in the left pane.

4. In the search box, type in the desired operating system. For example, if `rhel-6.6 HVM` is typed in the search box, the search results show the versions of RHEL v6.6 that support HVM. Select the highest GA number in order to use the latest release of v6.6 supporting HVM:

Step 1: Choose an Amazon Machine Image (AMI)[Cancel and Exit](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Quick Start

My AMIs

AWS Marketplace

Community AMIs

Operating system

- Amazon Linux
- Cent OS
- Debian
- Fedora
- Gentoo
- OpenSUSE
- Other Linux
- Red Hat
- SUSE Linux
- Ubuntu
- Windows

Architecture

- 32-bit
- 64-bit

Root device type

- EBS
- Instance store

420 results for "rhel-6.6 HVM" on AWS Marketplace
Partner software pre-configured to run on AWS

	RHEL-6.6_HVM_GA-20150601-x86_64-3-Hourly2-GP2 - ami-63a84227	Select
	Provided by Red Hat, Inc. Root device type: ebs Virtualization type: hvm	64-bit
	RHEL-6.6_HVM_GA-20141017-x86_64-1-Hourly2-GP2 - ami-69ccd92c	Select
	Provided by Red Hat, Inc. Root device type: ebs Virtualization type: hvm	64-bit
	RHEL-6.6_HVM_GA-20150128-x86_64-1-Hourly2-GP2 - ami-8a5048cf	Select
	Provided by Red Hat, Inc. Root device type: ebs Virtualization type: hvm	64-bit
	RHEL-6.6_HVM_GA-20150319-x86_64-1-Hourly2-GP2 - ami-f3a243b7	Select
	Provided by Red Hat, Inc. Root device type: ebs Virtualization type: hvm	64-bit
	RHEL-6.6_HVM_GA-SAP-20150430-x86_64-2-Hourly2-GP2-b676039c-a4f8-4be7-9866-c804b1ade684-ami-905a54f8.2 - ami-3311fa77	Select
	Provided by Red Hat, Inc. Root device type: ebs Virtualization type: hvm	64-bit
	RHEL-6.6_HVM_GA-SAP-20150430-x86_64-2-Hourly2-GP2-3c37fea0-fcf1-4aeb-8b72-ed7b333f22a9-ami-905a54f8.2 - ami-fd03eab9	Select
	Provided by Red Hat, Inc. Root device type: ebs Virtualization type: hvm	64-bit

5. Click **Select** for the AMI version you choose.

Using the AWS CLI

The second way to find AMIs is to generate a list of suitable AMIs with the AWS CLI. For example, to generate a list of RHEL 64-bit AMIs using the AWS CLI, run the query below.

Prerequisite: Install the AWS CLI and follow the setup instructions to be able to run commands.

```
aws ec2 describe-images \
--output table \
--query 'Images[*].[VirtualizationType,Name,ImageId]' \
--owners 309956199498 \
--filters \
Name=root-device-type,Values=ebs \
Name=image-type,Values=machine \
Name=is-public,Values=true \
Name=hypervisor,Values=xen \
Name=architecture,Values=x86_64
--
```

Creating an EC2 Instance for Cloudera Director

On AWS, Cloudera Director requires a dedicated Amazon EC2 instance in the same subnet that can access new instances on the private network.

To create the instance, follow these steps:

1. In the AWS Management Console, select **EC2** from the **Services** navigation list box in the desired region.
2. Click the **Launch Instance** button in the Create Instance section of the EC2 dashboard.
3. Select the desired AMI for your Cloudera Director instance. Cloudera recommends that you choose from Community AMIs list and the latest release of the desired supported distribution. For supported distributions, see [Supported Software and Distributions](#) on page 16.
 - a. Select **Community AMIs** in the left pane.

- b. In the search box, type in the desired operating system. For example, if you type `rhel-6.6 HVM`, the search results show the versions of RHEL v6.6 that support HVM. Select the highest GA number in order to use the latest release of v6.6 supporting HVM.

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

The screenshot displays the AWS console interface for selecting an AMI. The search bar at the top contains the text "rhel-6.6 HVM". Below the search bar, there are navigation tabs for "Quick Start", "My AMIs", "AWS Marketplace", and "Community AMIs". The "Community AMIs" section is active, showing a list of search results for "rhel-6.6 HVM" on AWS Marketplace. The results list includes the following AMIs:

AMI ID	Provider	Root Device Type	Virtualization Type	Architecture
RHEL-6.6_HVM_GA-20150601-x86_64-3-Hourly2-GP2 - ami-63a84227	Red Hat, Inc.	ebs	hvm	64-bit
RHEL-6.6_HVM_GA-20141017-x86_64-1-Hourly2-GP2 - ami-69ccd92c	Red Hat, Inc.	ebs	hvm	64-bit
RHEL-6.6_HVM_GA-20150128-x86_64-1-Hourly2-GP2 - ami-8a5048cf	Red Hat, Inc.	ebs	hvm	64-bit
RHEL-6.6_HVM_GA-20150319-x86_64-1-Hourly2-GP2 - ami-f3a243b7	Red Hat, Inc.	ebs	hvm	64-bit
RHEL-6.6_HVM_GA-SAP-20150430-x86_64-2-Hourly2-GP2-b676039c-a4f8-4be7-9866-c804b1ade684-ami-905a54f8.2 - ami-3311fa77	Red Hat, Inc.	ebs	hvm	64-bit
RHEL-6.6_HVM_GA-SAP-20150430-x86_64-2-Hourly2-GP2-3c37fea0-fcf1-4aeb-8b72-ed7b333f22a9-ami-905a54f8.2 - ami-fd03eab9	Red Hat, Inc.	ebs	hvm	64-bit

The sidebar on the left provides filters for the search results:

- Operating system:** Amazon Linux, Cent OS, Debian, Fedora, Gentoo, OpenSUSE, Other Linux, Red Hat, SUSE Linux, Ubuntu, Windows.
- Architecture:** 32-bit, 64-bit.
- Root device type:** EBS, Instance store.

- c. Click **Select** for the AMI version you choose.
- Select the instance type for Cloudera Director. Cloudera recommends using `c3.large` or `c4.large` instances.
 - Click **Next: Configure Instance Details**.
 - Select the correct VPC and subnet.
 - The cluster launcher requires Internet access; from the **Auto-assign Public IP** list box, select **Enable**.
 - Use the default shutdown behavior, **Stop**.
 - Click the **Protect against accidental termination** checkbox.
 - (Optional) Click the IAM role drop-down list and select an IAM role.
 - Click **Next: Add Storage**. Cloudera Director requires 8 GB of storage at a minimum.
 - Click **Next: Tag Instance**. For the **Name** key, enter a name for the instance in the **Value** field. Optionally, click **Create Tag** to create additional tags for the instance (up to a maximum of 10 tags).

Step 5: Tag Instance

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. [Learn more](#) about tagging your Amazon EC2 resources.

Key (127 characters maximum)	Value (255 characters maximum)
Name	Cloudera Director ✕
<input type="button" value="Create Tag"/> (Up to 10 tags maximum)	

8. Click **Next: Configure Security Group**.
9. On the **Configure Security Group** page, use the procedure below and the table that follows to create a new security group or add ports to an existing group. If you already have a security group with the required ports for Cloudera Director, you can skip this step.
 - a. Select either **Create a new security group** or **Select an existing security group**. If you choose to edit an existing group, select the group you want to edit. If you choose to create a new group, enter a **Security group name** and **Description**.
 - b. Click the **Type** drop-down list, and select a protocol type. Type the port number in the **Port Range** field.
 - c. For each additional port needed, click the **Add Rule** button. Then click the **Type** drop-down list, select a protocol type, and type the port number in the **Port Range** field.

The following ports need to be open for the Cloudera Director EC2 instance:

Type	Protocol	Port Range	Source
SSH (22)	TCP (6)	22	0.0.0.0/0
ALL Traffic	ALL	ALL	<i>security_group_id</i>

10. Click **Review and Launch**. Scroll down to review the AMI details, instance type, and security group information, and then click **Launch**.
11. At the prompt for a key pair:
 - Select **Choose an existing key pair** and select the key pair you created in [Creating an SSH Key Pair](#) on page 21.
 - Click the check box that acknowledges that you have access to the private key.

Select an existing key pair or create a new key pair ✕

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Select a key pair

I acknowledge that I have access to the selected private key file (docuser.pem), and that without this file, I won't be able to log into my instance.

Cancel
Launch Instances

12 Click **Launch Instances**.

13 After the instance is created, note its public and private IP addresses.

Installing Cloudera Director Server on the EC2 Instance

Cloudera recommends that you install Cloudera Director server on your cloud provider within the subnet where you will create CDH clusters, since Cloudera Director must have access to the private IP addresses of the instances that it creates. To install Cloudera Director server, perform the tasks below. You must be either running as root or using sudo to perform these tasks.

1. SSH into the EC2 instance you created for Cloudera Director. If you have VPN or AWS Direct Connect, SSH to your private IP address. Otherwise use your public IP address.
 - The default username for RHEL, CentOS, Oracle, and SLES is `ec2-user`.
 - The default username for Debian is `admin`.
 - The default username for Ubuntu is `ubuntu`.
2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 8, 7, and 6. For installation information, see [Java SE Downloads](#).
3. Download the Cloudera Director by running the correct command for your distribution.
 - For RHEL 6, CentOS 6, and Oracle 6:

```
wget http://archive.cloudera.com/director/redhat/6/x86_64/director/cloudera-director.repo
-O /etc/yum.repos.d/cloudera-director.repo
```

- For RHEL 5, CentOS 5, and Oracle 5:

```
wget http://archive.cloudera.com/director/redhat/5/x86_64/director/cloudera-director.repo
-O /etc/yum.repos.d/cloudera-director.repo
```

- For SLES:

```
zypper addrepo -f
http://archive.cloudera.com/director/sles/11/x86_64/director/cloudera-director.repo
```

- For Debian:

```
wget
http://archive.cloudera.com/director/debian/wheezy/amd64/director/cloudera-director.list
-O /etc/apt/sources.list.d/cloudera-director.list
```

- For Ubuntu 12.04 (Precise Pangolin):

```
wget
http://archive.cloudera.com/director/ubuntu/precise/amd64/director/cloudera-director.list
-O /etc/apt/sources.list.d/cloudera-director.list
```

- For Ubuntu 14.04 (Trusty Tahr):

```
wget
http://archive.cloudera.com/director/ubuntu/trusty/amd64/director/cloudera-director.list
-O /etc/apt/sources.list.d/cloudera-director.list
```

4. Add the signing key.

- For RHEL 6, CentOS 6, and Oracle 6 this step is not required. Continue to the next step.
- For RHEL 5, CentOS 5, and Oracle 5 this step is not required. Continue to the next step.
- For SLES this step is not required. Continue to the next step.
- For Debian, run the following command:

```
curl -s http://archive.cloudera.com/director/debian/wheezy/amd64/director/archive.key
| sudo apt-key add -
```

- For Ubuntu 12.04 (Precise Pangolin), run the following command:

```
curl -s http://archive.cloudera.com/director/ubuntu/precise/amd64/director/archive.key
| sudo apt-key add -
```

- For Ubuntu 14.04 (Trusty Tahr), run the following command:

```
curl -s http://archive.cloudera.com/director/ubuntu/trusty/amd64/director/archive.key
| sudo apt-key add -
```

5. Install Cloudera Director server by running the correct command for your distribution.

- For RHEL 6, CentOS 6, and Oracle 6:

```
yum install cloudera-director-server
```

- For RHEL 5, CentOS 5, and Oracle 5:

```
yum install cloudera-director-server
```

- For SLES:

```
zypper install cloudera-director-server
```

Getting Started with Cloudera Director

- For Debian, Ubuntu 12.04 (Precise Pangolin), and Ubuntu 14.04 (Trusty Tahr):

```
apt-get update
apt-get install cloudera-director-server
apt-get install oracle-j2sdk1.7
```

6. Start the Cloudera Director server by running the following command:

```
service cloudera-director-server start
```

7. Save the existing iptables rule set and disable the firewall:

- For RHEL 6, CentOS 6, and Oracle 6:

```
iptables-save > ~/firewall.rules
chkconfig iptables off
/etc/init.d/iptables stop
```

- For RHEL 5, CentOS 5, and Oracle 5:

```
iptables-save > ~/firewall.rules
chkconfig iptables off
/etc/init.d/iptables stop
```

- For SLES:

```
iptables-save > ~/firewall.rules
chkconfig SuSEfirewall2_setup off
rcSuSEfirewall2 stop
```

- For Debian:

```
iptables-save > ~/firewall.rules
chkconfig iptables off
/etc/init.d/iptables stop
```

- Ubuntu 12.04 (Precise Pangolin), and Ubuntu 14.04 (Trusty Tahr):

```
iptables-save > ~/firewall.rules
service ufw stop
```

You are now ready to install Cloudera Manager and CDH on the Cloudera Director server.

Viewing Cloudera Director Logs

To help you troubleshoot problems, you can view the Cloudera Director server log file. There is one server log file for all clusters. The log file can be found in the following location:

```
/var/log/cloudera-director-server/application.log
```

Deploying Cloudera Manager and CDH on AWS

To deploy Cloudera Manager and CDH on an AWS EC2 instance, you begin by creating an environment. The environment defines common settings, like region and key pair, that Cloudera Director uses with AWS. While creating an environment, you are also prompted to deploy its first cluster.

To create an environment:

1. Open a web browser and navigate to the private IP address of the instance you created in [Creating an EC2 Instance for Cloudera Director](#) on page 23. Include port 7189 in the address. For example:

```
http://192.0.2.0:7189
```

2. In the **Cloudera Director** login screen, enter `admin` in both the **Username** and the **Password** fields.
3. In the Cloudera Director **Welcome** screen, click **Let's get started**.

This opens a wizard for adding an environment, adding Cloudera Manager, and adding a CDH cluster.

4. In the **Add Environment** screen:

- Enter a name in the **Environment Name** field.
- Select **Amazon Web Services (AWS)** from the **Cloud provider** field.
- In the **Advanced Options** area, enter your AWS credentials in the **Access key ID** and **Secret access key** fields.

GENERAL INFORMATION

Environment name ?

Cloud provider ?

▼ **Advanced Options**

Make credentials available as cluster client configurations ?

Access key ID ?

Secret access key ?

- In the **Advanced Options** for **EC2**, select the same **EC2 region** that your EC2 Cloudera Director instance was

EC2

▼ **Advanced Options**

Associate public IP addresses ?

IAM endpoint ?

EC2 region ?

EC2 region endpoint ?

created in.

- In the **SSH Credentials** section:
 - Enter `ec2-user` in the **Username** field.
 - Copy the SSH private key you created in [Creating an EC2 Instance for Cloudera Director](#) on page 23 in the **Private key** field.

SSH CREDENTIALS

Username ?

Private key File Upload Direct Input ?

5. Click **Continue** to add Cloudera Manager.

6. In the **Add Cloudera Manager** screen:

- Enter a name for this deployment of Cloudera Manager in the **Cloudera Manager name** field.
- In the **Instance Template** field, select **Create New Instance Template**.

The **Instance Template** modal screen displays.

Add Cloudera Manager

Environment TESTENV01

Cloudera Manager name ?

Instance Template ?

Database Server ?

7. In the **Instance Template** modal screen, do the following:

- In the **Instance Template name** field, enter a name for the template.
- In the **Instance type** field, select **m4.large** or **m4.xlarge**.
- In the **Image (AMI) ID** field, enter the ID for the Amazon machine image (AMI) you chose in [Creating an EC2 Instance for Cloudera Director](#) on page 23, or find another AMI with the same operating system.
- In the **Tags** field, add one or more tags to associate with the instance.
- In the **Security group IDs** field, enter the security group ID you set up in [Creating a New Security Group](#) on page 21.
- In the **VPC subnet ID** field, enter the ID of the VPC subnet you set up in [Creating Your Subnet\(s\)](#) on page 20.
- Click **Save changes**.

Instance Template ✕

Instance Template name

Instance type ?

Image (AMI) ID ?

Tags

Name	Value		
<input type="text" value="Name"/>	<input type="text" value="test-instance"/>	-	+

Security group IDs - + ?

VPC subnet ID ?

[> Advanced Options](#)

[Cancel](#) [Save changes](#)

8. In the **Add Cloudera Manager** screen, click **Continue**.
9. At the **Confirmation** prompt, click **OK** to begin adding a cluster.
10. On the **Add Cluster** screen:
 - Enter a name for the cluster in the **Cluster name** field.
 - Select the version of CDH to deploy in the **Version** field.
 - In the **Services** section, select the services you want to install.
 - In the **Instance groups** area, select the instance template you created earlier for each group and the number of instances you want.

Instance groups

Name ?	Roles	Instance Template	Instance Count
<input type="text" value="masters"/>	Edit Roles	<input type="text" value="TEST-TEMPLATE"/> Edit	<input type="text" value="1"/>
<input type="text" value="workers"/>	Edit Roles	<input type="text" value="TEST-TEMPLATE"/> Edit	<input type="text" value="5"/>
<input type="text" value="gateway"/>	Edit Roles	<input type="text" value="TEST-TEMPLATE"/> Edit	<input type="text" value="1"/>
Add Group			

11. Click **Continue**.

12 At the **Confirmation** prompt, click **OK** to deploy the cluster. Cloudera Director displays a status screen.

Status

TESTCLUSTER01 Bootstrapping

7 / 30

REQUESTING 7 INSTANCE(S) IN 3 GROUP(S)

1. Starting
2. Starting
3. Starting

13 When the cluster is ready, click **Continue**.

Cloudera Director Client

The Cloudera Director client works well for proof-of-concept demonstrations, development work, and infrequent usage. Deployment through the Cloudera Director client involves installing on an instance, editing a configuration file, and running Cloudera Director from the command line. Cloudera Director client installation, configuration, and use are described in the following topics.

Installing Cloudera Director Client

To install Cloudera Director client, perform the following tasks.



Important: Cloudera Director requires a JDK. For more information, see [Supported Software and Distributions](#) on page 16.

1. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 8, 7, and 6. For installation information, see [Java SE Downloads](#).
2. Download Cloudera Director by running the correct command for your distribution.

- For RHEL 6, CentOS 6, and Oracle 6:

```
wget http://archive.cloudera.com/director/redhat/6/x86_64/director/cloudera-director.repo
-O /etc/yum.repos.d/cloudera-director.repo
```

- For RHEL 5, CentOS 5, and Oracle 5:

```
wget http://archive.cloudera.com/director/redhat/5/x86_64/director/cloudera-director.repo
-O /etc/yum.repos.d/cloudera-director.repo
```

- For SLES:

```
zypper addrepo -f
http://archive.cloudera.com/director/sles/11/x86_64/director/cloudera-director.repo
```

- For Debian:

```
wget
http://archive.cloudera.com/director/debian/wheezy/amd64/director/cloudera-director.list
-O /etc/apt/sources.list.d/cloudera-director.list
```

- For Ubuntu 12.04 (Precise Pangolin):

```
wget
http://archive.cloudera.com/director/ubuntu/precise/amd64/director/cloudera-director.list
-O /etc/apt/sources.list.d/cloudera-director.list
```

- For Ubuntu 14.04 (Trusty Tahr):

```
wget
http://archive.cloudera.com/director/ubuntu/trusty/amd64/director/cloudera-director.list
-O /etc/apt/sources.list.d/cloudera-director.list
```

3. Add the signing key.

- For RHEL 6, CentOS 6, and Oracle 6 this step is not required. Continue to the next step.
- For RHEL 5, CentOS 5, and Oracle 5 this step is not required. Continue to the next step.

- For SLES this step is not required. Continue to the next step.
- For Debian, run the following command:

```
curl -s http://archive.cloudera.com/director/debian/wheezy/amd64/director/archive.key | sudo apt-key add -
```

- For Ubuntu 12.04 (Precise Pangolin), run the following command:

```
curl -s http://archive.cloudera.com/director/ubuntu/precise/amd64/director/archive.key | sudo apt-key add -
```

- For Ubuntu 14.04 (Trusty Tahr), run the following command:

```
curl -s http://archive.cloudera.com/director/ubuntu/trusty/amd64/director/archive.key | sudo apt-key add -
```

4. Install Cloudera Director client by running the correct command for your distribution.

- For RHEL 6, CentOS 6, and Oracle 6:

```
yum install cloudera-director-client
```

- For RHEL 5, CentOS 5, and Oracle 5:

```
yum install cloudera-director-client
```

- For SLES:

```
zypper install cloudera-director-client
```

- For Debian, Ubuntu 12.04 (Precise Pangolin), and Ubuntu 14.04 (Trusty Tahr):

```
apt-get install cloudera-director-client
```

You are now ready to configure the Cloudera Director Client.

Provisioning a Cluster on AWS

The configuration file contains information Cloudera Director needs to operate and settings that define your cluster.

Sample configuration files are found either in `/usr/lib64/cloudera-director/client` or `/usr/lib/cloudera-director/client`, depending on the operating system you are using. Copy the sample files to your home directory before editing them.

To modify the configuration file:

1. Rename the `aws.simple.conf` file to `cluster.conf`. For advanced cluster configuration, use `aws.reference.conf`.



Note: The configuration file must use the `.conf` file extension.

2. Open `cluster.conf` with a text editor.
3. Configure the basic settings:
 - **name** - change to something that makes the cluster easy to identify.
 - **id** - leave this set to `aws`.

- **accessKeyId** - AWS access key ID. Make sure the value is enclosed in double quotes.
- **secretAccessKey** - AWS secret access key. Make sure the value is enclosed in double quotes.
- **region** - specify the region (for example, us-west-2).
- **keyName** - specify the name of the key pair used to start the cluster launcher. Key pairs are region-specific. For example, if you create a key pair (or import one you have created) in US-West-2, it will not be available in US-West-1. For information on creating key pairs in Amazon EC2 or importing existing key pairs, see [Amazon EC2 Key Pairs](#).
- **subnetId** - ID of the subnet that you noted earlier.
- **securityGroupsIds** - ID of the security group that you noted earlier. Use the ID of the group, not the name (for example, sg-b139d3d3, not default).
- **instanceNamePrefix** - enter the prefix to prepend to each instance's name.
- **image** - specifies the AMI to use. Cloudera recommends Red Hat Enterprise Linux 6.4 (64bit). To find the correct AMI for the selected region, visit the Red Hat AWS Partner page.



Note: If you use your own AMI, make sure to disable any software that prevents the instance from rebooting during the deployment of the cluster.

4. Configure the following cluster settings:

- You can only use Cloudera Manager 5. No changes are needed for repository and repository key URLs and you must set the parcel repositories to match the CDH and Impala versions you plan to install.
- Specify services to start on the cluster. For a complete list of allowed values, see the [Cloudera Manager API Service Types](#).



Note: Include Flume in the list of services only when customizing role assignments. See the configuration file (`aws.reference.conf`) included in the Cloudera Director download for examples on how to configure customized role assignments. If Flume is required, it should be excluded from the list of services in the configuration file and added as a service using Cloudera Manager UI or API after the cluster is deployed. When adding Flume as a service, you must assign Flume agents (which Cloudera Manager does not do automatically).

- Specify the number of instances in the cluster.

5. Save the file and exit.



Note: If your root disk drive is larger than all the other drives on the machine, Cloudera Manager automatically installs HDFS on the root drive. You can change this behavior with an explicit override in the `configs {}` block within the `cluster {}` section of the configuration file.

Running Cloudera Director Client

After you modify the configuration file, you can run Cloudera Director client. There are two ways of running the Cloudera Director client:

- In standalone mode, using the `bootstrap` command. Clusters created using the `bootstrap` command cannot be managed using the Cloudera Director UI. The information below on this page concerns running the client in standalone mode.
- If you already have a server, you can run the client against the server using the commands `bootstrap-remote` and `terminate-remote`. Only clusters created with the `bootstrap-remote` command can be managed using the Cloudera Director UI. For more information on using the client to deploy clusters on the server, see [Submitting a Cluster Configuration File](#).



Note: If you are restarting Cloudera Director client, you are prompted to resume from where the client stopped or start over. If you made changes to the configuration file between deployments, or if you need to start the run from scratch, you should start over.

1. From the cluster launcher, enter the following:

```
[ec2-user@ip-10-1-1-18 cloudera-director-1.1.0]$ cloudera-director bootstrap cluster.conf
```

Cloudera Director displays output similar to the following:

```
Installing Cloudera Manager ...
* Starting ... done
* Requesting an instance for Cloudera Manager ..... done
* Inspecting capabilities of 10.1.1.194 ..... done
* Normalizing 10.1.1.194 ..... done
* Installing python (1/4) .... done
* Installing ntp (2/4) .... done
* Installing curl (3/4) .... done
* Installing wget (4/4) ..... done
* Installing repositories for Cloudera Manager ..... done
* Installing jdk (1/5) ..... done
* Installing cloudera-manager-daemons (2/5) ..... done
* Installing cloudera-manager-server (3/5) ..... done
* Installing cloudera-manager-server-db-2 (4/5) ..... done
* Installing cloudera-manager-agent (5/5) .... done
* Starting embedded PostgreSQL database ..... done
* Starting Cloudera Manager server ..... done
* Waiting for Cloudera Manager server to start .... done
* Configuring Cloudera Manager ..... done
* Starting Cloudera Management Services ..... done
* Inspecting capabilities of 10.1.1.194 ..... done
* Done ...
Cloudera Manager ready.
Creating cluster C5-Sandbox-AWS ...
* Starting ... done
* Requesting 3 instance(s) ..... done
* Inspecting capabilities of new instance(s) ..... done
* Running basic normalization scripts ..... done
* Registering instance(s) with Cloudera Manager .... done
* Waiting for Cloudera Manager to deploy agents on instances ... done
* Creating CDH5 cluster using the new nodes ..... done
* Downloading CDH-5.4.0-1.cdh5.4.0.p0.26 parcel ..... done
* Distributing CDH-5.4.0-1.cdh5.4.0.p0.26 parcel ... done
* Activating CDH-5.4.0-1.cdh5.4.0.p0.26 parcel ..... done
* Done ...
Cluster ready.
```



Note: If you have a large root disk partition or if you are using a hardware virtual machine (HVM) AMI, the instances can take a long time to reboot. Cloudera Manager can take 20-25 minutes to become available.

2. To monitor Cloudera Director, log in to the cluster launcher and view the application log:

```
$ ssh ec2-user@54.186.148.151
Last login: Tue Mar 18 20:33:38 2014 from 65.50.196.130
[ec2-user@ip-10-1-1-18]$ tail -f ~/.cloudera-director/logs/application.log
[...]
```



Note: If you have deployment issues and need help troubleshooting, be careful when distributing the `state.h2.db` or `application.log` files. They contain sensitive information, such as your AWS keys and SSH keys.

Using the Command Line Interface

The command-line interface (CLI) includes commands and options for running Cloudera Director locally or for bootstrapping or terminating Cloudera Director on a remote server.

Local commands

The commands in this table can be used when running Cloudera Manager locally (in standalone mode):

Command	Description	Options
<code>bootstrap</code>	Bootstraps an environment, deployment, and cluster locally (in standalone mode).	<code>lp.bootstrap.resume policy=interactive resume restart</code> <ul style="list-style-type: none"> If progress was already made bootstrapping, this option determines if the process will automatically resume (<code>resume</code>), start over from scratch (<code>restart</code>), or ask the user (<code>interactive</code>). The default value is <code>interactive</code>.
<code>status</code>	Reports status on various entities, including deployment, cluster, Cloudera Manager instance, and cluster instances.	
<code>terminate</code>	Terminates a cluster and deployment locally (in standalone mode).	<code>lp.terminate.assumeYes=true false</code> <ul style="list-style-type: none"> This property determines if the user must explicitly confirm termination (<code>false</code>) or if confirmation is assumed (<code>true</code>). The default value is <code>false</code>.
<code>update</code>	Updates an environment, deployment, and cluster locally (in standalone mode).	
<code>validate</code>	Validates a configuration locally (in standalone mode).	<code>lp.validate.dumpTemplates=true false</code> <ul style="list-style-type: none"> If <code>true</code>, prints out parsed configuration information. The default value is <code>false</code>.

Remote commands

The following CLI commands can be used to bootstrap or terminate Cloudera Director on a remote server:

Command	Description	Option
<code>bootstrap-remote</code>	Bootstraps an environment, deployment, and cluster on a remote server.	<code>lp.remote.hostAndPort=host[:port]</code> <ul style="list-style-type: none"> Default value: <code>localhost:7189</code>

Command	Description	Option
		<code>lp.remote.username=Cloudera Director server username</code> <code>lp.remote.password=Cloudera Director server password</code>
<code>terminate-remote</code>	Terminates a cluster and deployment on a remote server.	<code>lp.remote.hostAndPort=host[:port]</code> <ul style="list-style-type: none"> • Default value: <code>localhost:7189</code> <code>lp.remote.username=Cloudera Director server username</code> <code>lp.remote.password=Cloudera Director server password</code> <code>lp.remote.terminate.assumeYes=true false</code> <ul style="list-style-type: none"> • This property determines if the user must explicitly confirm termination (<code>false</code>) or if confirmation is assumed (<code>true</code>). The default value is <code>false</code>.

Connecting to Cloudera Manager

After the cluster is ready, log in to Cloudera Manager and access the cluster.

To access Cloudera Manager:

1. Use the status command to get the host IP address of Cloudera Manager:

```
$ cloudera-director status cluster.conf
```

Cloudera Director displays output similar to the following:

```
Cloudera Launchpad 1.0.0 initializing ...

Cloudera Manager:
* Instance: 10.0.0.110 Owner=wintermute,Group=manager
* Shell: ssh -i /root/.ssh/launchpad root@10.0.0.110

Cluster Instances:
* Instance 1: 10.0.0.39 Owner=wintermute,Group=master
* Shell 1: ssh -i /root/.ssh/launchpad root@10.0.0.39

* Instance 2: 10.0.0.148 Owner=wintermute,Group=slave
* Shell 2: ssh -i /root/.ssh/launchpad root@10.0.0.148

* Instance 3: 10.0.0.150 Owner=wintermute,Group=slave
* Shell 3: ssh -i /root/.ssh/launchpad root@10.0.0.150

* Instance 4: 10.0.0.147 Owner=wintermute,Group=slave
* Shell 4: ssh -i /root/.ssh/launchpad root@10.0.0.147

* Instance 5: 10.0.0.149 Owner=wintermute,Group=slave
* Shell 5: ssh -i /root/.ssh/launchpad root@10.0.0.149

* Instance 6: 10.0.0.151 Owner=wintermute,Group=slave
* Shell 6: ssh -i /root/.ssh/launchpad root@10.0.0.151

* Instance 7: 10.0.0.254 Owner=wintermute,Group=gateway
* Shell 7: ssh -i /root/.ssh/launchpad root@10.0.0.254
```

```
* Instance 8: 10.0.0.32 Owner=wintermute,Group=master
* Shell 8: ssh -i /root/.ssh/launchpad root@10.0.0.32

* Instance 9: 10.0.0.22 Owner=wintermute,Group=master
* Shell 9: ssh -i /root/.ssh/launchpad root@10.0.0.22

Launchpad Gateway:
* Gateway Shell: ssh -i /path/to/launchpad/host/keyName.pem -L 7180:10.0.0.110:7180 -L
7187:10.0.0.110:7187 root@ec2-54-77-57-3.eu-west-1.compute.amazonaws.com

Cluster Consoles:
* Cloudera Manager: http://localhost:7180
* Cloudera Navigator: http://localhost:7187
```

In this example, the host IP address is 10.0.0.110.

2. Change to the directory where your `keyfile.pem` file is located. Then, route the connection over SSH:

```
$ ssh -L 7180:cm-host-private-ip:7180 ec2-user@cm-host-public-ip
# go to http://localhost:7180 in your browser and login with admin/admin
```



Note: If you get a permission error, add the `.pem` file from the command line:

```
$ ssh -i <keyfile.pem> -L 7180:cm-host-private-ip:7180
ec2-user@cm-host-public-ip
```

3. Open a web browser and enter `http://localhost:7180` to connect to Cloudera Manager. Use `admin` as both the username and password.
4. Add any additional services to the cluster. The CDH 5 parcel was already distributed by Cloudera Director.

Modifying a Cluster with Cloudera Director Client

This section describes how to make changes to the cluster through Cloudera Director, using the client and the configuration file.

Growing or Shrinking a Cluster

After launching a cluster, you can add or remove instances:

1. Open the `cluster.conf` file that you used to launch the cluster.
2. Change the value for the type of instance you want to change. For example, the following increases the number of workers to 15:

```
workers {
  count: 15
  minCount: 5

  instance: ${instances.hs18} {
    tags {
      group: worker
    }
  }
}
```

3. Enter the following command:

```
cloudera-director update cluster.conf
```

Cloudera Director increases the number of worker instances.

4. Assign roles to the new master instances through Cloudera Manager. Cloudera Director does not automatically assign roles.

Managing Cloudera Manager Instances with Cloudera Director Server

The Cloudera Director server is designed to run in a centralized setup, managing multiple Cloudera Manager instances and CDH clusters, with multiple users and user accounts. The server works well for launching and managing large numbers of clusters in a production environment. Cloudera Director server installation, configuration, and use are described in the following topics.

Submitting a Cluster Configuration File

In Cloudera Director, you can deploy clusters in two ways:

- Through the Cloudera Director server UI.
- Through the Cloudera Director client, which you can use to send a configuration file that the server uses for cluster deployment. The configuration file provides advanced options not currently available in the server UI.

This section describes the second of these ways, using the Cloudera Director client to submit a configuration file. The configuration file will be applied to the cluster and managed by the Cloudera Director server.

When you submit a cluster configuration from a Cloudera Director client to the Cloudera Director server, all communications are transmitted in the clear (including the AWS credentials). If the client and server communicate over the Internet, use a VPN for security.



Note: If you create tags in the configuration file for AWS instance metadata or for service or role configurations, special characters, such as are required by the HOCON format, must be quoted, including periods. An example a tag value that would require quoting is "company:department:team". See the AWS documentation for information about which special characters are supported on these cloud platforms in instance metadata tags.

To submit a cluster configuration file to the Cloudera Director server, follow these steps:

1. Create a configuration file. See [Provisioning a Cluster on AWS](#) on page 34.
2. Install the latest version of the Cloudera Director client from the [Cloudera Director Download Page](#).
3. Unzip the Cloudera Director client.
4. Change to the client directory and enter the following:

```
cloudera-director bootstrap-remote myconfig.conf --lp.remote.username=admin
--lp.remote.password=admin --lp.remote.hostAndPort=host:port
```

myconfig.conf is the name of your configuration file, *admin* is the default value for both the username and password for the Admin account (enter your actual values), *host* is the name or IP address of the host on which Cloudera Director server is running, and *port* is the port on which it is listening.

The Cloudera Director client provides deployment status.

Deploying Clusters in an Existing Environment

If you already configured an environment, you can easily deploy a new cluster:

1. Log in to Cloudera Director. For example, <http://example.com:7189>.
2. Click **Add Cluster**, and then select an environment from the **Environment** list box. .
3. Select a Cloudera Manager from the **Cloudera Manager** list box.
4. To clone an existing cluster, select **Clone from existing** and select a cluster. To specify cluster settings, select **Create from scratch**.
5. Enter a name for the cluster in the **Cluster name** field, and select the version of CDH to deploy in the **Version** field.

6. Select the type of cluster to deploy from **Services**.
7. Select the numbers of masters, workers, and gateways to deploy. Then, select an instance template for each or create one or more new templates.
8. When you are finished, click **Continue**. When prompted for confirmation, click **OK** to confirm.

Cloudera Director begins deploying the cluster.



Note: If your root disk drive is larger than all the other drives on the machine, Cloudera Manager automatically installs HDFS on the root drive.

Cloudera Manager Health Information

The following Cloudera Manager health information is available through Cloudera Director server:

- Host health
- Service health
- Cluster health

The health value is displayed in the **Status** column for each entity, when health information is available. Possible health values are:

- **Disabled** - Health collection has been disabled on Cloudera Manager.
- **Not Available** - Cloudera Director does not currently have health information, or a health has "expired."
- **Bad** - Cloudera Manager reports the health as bad.
- **Concerning** - Cloudera Manager reports the health as concerning.
- **Good** - Cloudera Manager reports the health as good.

You can configure the health cache with the following settings in the `application.properties` file:

- `lp.cache.health.pollingRateInMilliseconds` - How often the Cloudera Director server polls Cloudera Manager for health information. The default value is 30,000 ms (30 seconds). To disable health collection, set `lp.cache.health.pollingRateInMilliseconds` to 0.
- `lp.cache.health.numberOfHealthCacheExecutorThreads` - The number of threads used to simultaneously request health information from Cloudera Manager. the default value is 5.
- `lp.cache.health.expirationMultiplier` - Used to determine if a health value is stale. If the health value has not been updated in `pollingRateInMilliseconds * expirationMultiplier` milliseconds, then the health value is considered stale and is reported to the UI as NOT_AVAILABLE. Using the default settings, for example, if health has not been reported in $2 * 30,000$ milliseconds = 60 seconds, it becomes stale. The default value is 2.



Note: Cloudera Manager health is collected by Cloudera Director server only, not by Cloudera Director client.

Opening Cloudera Manager

After deploying a cluster, you can manage it using Cloudera Manager:

1. Log in to Cloudera Director. For example, <http://example.com:7189>.

Cloudera Director opens with a list of clusters.

2. Locate the cluster to manage and click its Cloudera Manager. The link is available when Cloudera Manager is ready.
3. On the Cloudera Manager Login page, enter your credentials and click **Login**.

Cloudera Manager opens.

Adding HDFS DataNodes to a Cluster

You can use Cloudera Director to increase the number of HDFS DataNodes in a cluster:

1. Log in to Cloudera Director at `http://director-server-hostname:7189`.
Cloudera Director opens with a list of clusters.
2. If the cluster has a status of **Ready**, click the **Actions** list box to the right of the target cluster and select **Modify Cluster**.
The Modify Cluster page appears, displaying the number of gateways, workers, and masters.
3. On the Modify Cluster page, click **Edit** and increase the number of workers and gateways to the desired size.



Note: Cloudera recommends rebalancing the cluster through Cloudera Manager if you increase the number of HDFS DataNodes by 30% or more.

Removing or Repairing Hosts in a Cluster

Cloudera Director can remove or repair hosts in a cluster as described in the following sections.

Removing Hosts from a Cluster

Cloudera Director can reduce the size of a cluster by removing hosts that contain worker or gateway roles.

To shrink a cluster:

1. Log in to Cloudera Director at `http://director-server-hostname:7189`.
Cloudera Director opens with a list of clusters.
2. If the cluster has a status of **Ready**, click the **Actions** list box to the right of the cluster to shrink and select **Modify Cluster**.
The Modify Cluster page appears, displaying the number of gateways, workers, and masters.
3. To remove all instances for a role type:
 - On the Modify Cluster page, click **Delete Group**.To remove individual instances (that is, individual hosts for a role instance group):
 - Click **Edit** next to the instance count for workers or gateways, select the hosts you want to remove, and click the **Delete** button above the list of instances. The instances you selected display an action status of **To be deleted**.
4. Click **OK** to continue, **Reset** to unselect the selected instances and make a new selection, or **Cancel** to stop editing the instance group without making any changes.
5. Click **Continue** to confirm and delete the selected instances.

**Note:**

- It is important to maintain the number of HDFS DataNode role instances at or above the HDFS replication factor configured for the cluster. By default, Cloudera recommends a replication factor of three.
- Reducing or repairing the instance count to a running level that is below the replication factor can cause Cloudera Manager to fail to decommission DataNodes. In this case, Cloudera Director may stop functioning properly. If this happens, abort the decommission command in Cloudera Manager.
- If the instance count is below the replication factor, reduce the replication factor before attempting a repair.
- Cloudera recommends rebalancing the cluster through Cloudera Manager if you reduce the number of DataNodes by 30% or more.

Repairing Hosts in a Cluster

To repair hosts in a cluster:

1. Log in to Cloudera Director at `http://director-server-hostname:7189`.

Cloudera Director opens with a list of clusters.

2. If the cluster has a status of **Ready**, click the **Actions** list box to the right of the cluster to repair and select **Modify Cluster**.

The Modify Cluster page appears, displaying the number of gateways, workers, and masters.

3. Click **Edit** next to the instance count for workers or gateways you want to repair, and select the hosts you want to repair.
4. Click the **Repair** button above the list of instances. The hosts you selected display an action status of **To be repaired**.
5. Click **OK** to continue, **Reset** to unselect the selected instances and make a new selection, or **Cancel** to stop editing the instance group without making any changes.
6. Click **Continue** to confirm and repair the selected instances.

Terminating a Cluster

You can terminate a cluster at any time using either the UI or the CLI.

Terminating a Cluster with the UI

To terminate a cluster with the UI:

1. Log in to Cloudera Director. For example, `http://cloudera_director_host:7189`.

Cloudera Director opens with a list of clusters.

2. Click the Actions dropdown arrow for the cluster you want to terminate and click **Terminate**.
3. In the confirmation dialog box, click **Terminate** to terminate the cluster.

Terminating a Cluster with the CLI

For information on terminating a cluster with the CLI, see the section on the `terminate-remote` command in [Using the Command Line Interface](#).

Starting and Stopping the Cloudera Director Server

Although you can stop and start Cloudera Director at any time, you should terminate any running clusters first.

To start or stop the server, enter the following:

```
$ sudo service cloudera-director-server [start | stop]
```

User Management

User roles control the actions a user can perform. There are currently two user roles:

- **Admin** - For administrative access. Has full access to Cloudera Director functionality, and can perform the following actions:
 - Add environments, Cloudera Manager instances, and clusters
 - Delete environments
 - Terminate Cloudera Manager and cluster instances
 - Review environments, Cloudera Manager instances, and clusters
 - Grow and shrink clusters
 - Add and delete users
 - Change user roles
 - Change passwords, including own password
- **Guest** - For read-only access.

On installation, the Cloudera Director server component includes one of each of the two kinds of user accounts:

- **admin** - Default password: `admin`
- **guest** - Default password: `guest`

Cloudera recommends that you change the passwords for these accounts after installing the server. User accounts can be created, deleted, enabled, or disabled. A disabled user account cannot log in or perform any Cloudera Director actions.

User account data is kept in the Cloudera Director database. You can define new user accounts for Cloudera Director with either the server UI or the API.

Managing Users with the Cloudera Director Web UI

You can perform the following user management operations through the Cloudera Director Web UI:

Create a User Account

To create a new user account, perform the following steps:

1. On the Cloudera Director home screen, click the dropdown menu in the upper right and click **Manage Users**.
2. Click the **Add User** button.
3. Enter a username and password for the new user, and select a role (Admin or Guest).
4. Click **Add User**.

Disable a User Account

To disable an existing user account, perform the following steps:

1. On the Cloudera Director home screen, click the dropdown menu in the upper right and click **Manage Users**.
2. Click the checkbox next to the user account you want to disable.
3. Click the dropdown menu for the user account in the **Actions** column and click **Disable User**.
4. Confirm that user you have disabled now appears grayed out on the Manage Users screen.

You can use the same procedure to enable a user account that is currently disabled. The Actions dropdown list displays the item **Enable User** for a user account that is currently disabled.

Change User Account Passwords

Users with the admin role can change any user's password. Guest users can change only their own password.

To change your own password, perform the following steps:

1. On the Cloudera Director home screen, click the dropdown menu in the upper right and click **Change password**.
2. Enter your current password, a new password, and the new password again to confirm.
3. Click **Save changes**.

To change another user's password, perform the following steps (using the required Admin role):

1. On the Cloudera Director home screen, click the dropdown menu in the upper right and click **Manage Users**.
2. Click the checkbox next to the user whose password you want to change.
3. Click the dropdown menu for the user account in the **Actions** column and click **Change password**.
4. Enter a new password and enter the password again to confirm.
5. Click **Save changes**.

Change a User's Role

An Admin user can change another user's role by performing the following steps:

1. On the Cloudera Director home screen, click the dropdown menu in the upper right and click **Manage Users**.
2. Click the checkbox next to the user whose role you want to change.
3. Click the dropdown menu for the user in the **Actions** column and click **Change role**.
4. Select the new role in the **Role** dropdown menu.
5. Click **Save changes**.

Delete a User Account

An Admin user can delete a user account by performing the following steps:

1. On the Cloudera Director home screen, click the dropdown menu in the upper right and click **Manage Users**.
2. Click the checkbox next to the user account you want to delete.
3. Click the dropdown menu for the user account in the **Actions** column and click **Delete**.
4. Click **Delete** to confirm.

Managing Users with the Cloudera Director API

Cloudera Director server has a REST service endpoint for user management, at `director-server-hostname:7189/api/v2/users`. You can perform the following user-management operations with the Cloudera Director API. They all use JSON for input data and response data.

REST method	Description
GET /api/v2/users	Lists all usernames.
POST /api/v2/users	Creates a new user account (Admin role required).
GET /api/v2/users/current	Gets account information on the currently logged-in user.
GET /api/v2/users/{username}	Gets account information on a user.
PUT /api/v2/users/{username}	Changes account information on a user.
DELETE /api/v2/users/{username}	Deletes an account (Admin role required)
PUT /api/v2/users/{username}/password	Changes an account password for Guests; old password required, and Guests can only change their own account.

Managing Cloudera Manager Instances with Cloudera Director Server

For information on managing users with the Cloudera Director API, see the server API documentation at *director-server-hostname:7189/api-console*. Expand the section labeled **users**.

Customization and Advanced Configuration

The topics in this section explain how to use some of the advanced features of Cloudera Director.

The Cloudera Director Configuration File

The Cloudera Director configuration file is used to launch a cluster through Cloudera Director client with the `bootstrap` command, or through the Cloudera Director server with the `bootstrap-remote` command.

Location of Sample Configuration Files

Sample configuration files are found either in `/usr/lib64/cloudera-director/client` or `/usr/lib/cloudera-director/client`, depending on the operating system you are using. Copy the sample files to your home directory before editing them.

Customizing the Configuration File

Copy the sample files to your home directory before editing them. Rename the `cloud_provider.simple.conf` file to `cluster.conf`. For advanced cluster configuration, use `cloud_provider.reference.conf`. The configuration file must use the `.conf` file extension. Open `cluster.conf` with a text editor.

The `cloud_provider.reference.conf` version of the configuration file includes advanced settings that are documented in comments within the file itself. Details on the specific settings in the file are not duplicated in this document.

Creating a Cloudera Manager and CDH AMI

You can reduce instance start times, and thereby cluster bootstrap times, by preloading the AMI with Cloudera Manager packages and CDH parcel files. For information on creating AMIs preloaded with Cloudera Manager packages and CDH parcels for use by Cloudera Director see [Cloudera Director preload creation script](#) on GitHub.



Note: If you are using an AMI that already has Cloudera Manager or CDH pre-loaded on it, you must override the repository in Cloudera Director by specifying a custom repository URL in the custom repository field. The version you specify in this URL override must match what is on your AMI, down to the three digits of the maintenance release. For example, if you have CDH 5.3.3 on the AMI, the repository you specify should be `/5.3.3` and not `/5.3` or `/5`.

Choosing an AMI

An Amazon Machine Image (AMI) specifies the operating system, architecture (32-bit or 64-bit), AWS Region, and virtualization type (Paravirtualization or HVM) for a virtual machine (also known as an instance) that you launch in AWS.



Important: Cloudera Director, CDH, and Cloudera Manager support only 64-bit Linux. For CDH and Cloudera Manager on Amazon EC2, Cloudera Director only supports RHEL and CentOS.

The virtualization type depends on the instance type that you use. After selecting an instance type based on the expected storage and computational load, check the [supported virtualization types](#). Then, identify the correct AMI based on [architecture, AWS Region, and virtualization type](#).

Finding Available AMIs

There are two ways of finding available AMIs:

- Using the [AWS Management Console](#).
- By generating a list of AMIs using the AWS CLI.

To generate a list of RHEL 64-bit AMIs using the AWS CLI, perform the following steps:

1. Install the AWS CLI.

```
$ sudo pip install awscli
```

2. Configure the AWS CLI.

```
$ aws configure
```

Follow the prompts. Choose any output format. The following example command defines "table" as the format.

3. Run the following query:

```
aws ec2 describe-images \  
--output table \  
--query 'Images[*].[VirtualizationType,Name,ImageId]' \  
--owners 309956199498 \  
--filters \  
  Name=root-device-type,Values=ebs \  
  Name=image-type,Values=machine \  
  Name=is-public,Values=true \  
  Name=hypervisor,Values=xen \  
  Name=architecture,Values=x86_64
```

AWS returns a table of available images in the region you configured.

Creating AWS Identity and Access Management (IAM) Policies

In AWS, you use IAM files to create policies that control access to resources in a VPC. Use the [AWS Policy Generator](#) to create the IAM file, keeping in mind the following requirements:

- For EC2, Cloudera Director requires permissions for the following methods:
 - CreateTags
 - DescribeAvailabilityZones
 - DescribeImages
 - DescribeInstanceStatus
 - DescribeInstances
 - DescribeKeyPairs
 - DescribePlacementGroups
 - DescribeRegions
 - DescribeSecurityGroups
 - DescribeSubnets
 - RunInstances
 - TerminateInstances
- To validate the templates used for EC2 instance creation, Cloudera Director requires permissions for the following IAM methods:
 - GetInstanceProfile

- PassRole
- To create RDS database servers for persistence on demand, Cloudera Director requires permissions for the following methods:
 - CreateDBInstance
 - DeleteDBInstance
 - DescribeDBInstances
- With Cloudera Director 1.5 and higher, Cloudera Director requires permissions for the following method:
 - DescribeDBSecurityGroups

This permission is required because, beginning with version 1.5, Cloudera Director includes early validation of RDS credentials at the time of creating or updating an environment, whether or not RDS database servers will be used.

Example IAM Policy

The following example IAM policy shows the format to use with Cloudera Director. Your Amazon Resource Name (ARN) will be different.



Note: If Cloudera Director does not have the complete set of permissions it needs, an authorization failure may occur. In that event, AWS will return an authorization failure message, which may help with troubleshooting by providing details about the authorization failure. Authorization failure messages are normally encoded for security purposes. The permission shown in the last section of the example IAM policy below (beginning "Sid": "directorSts") enables Cloudera Director to decode authorization failure messages. Before adding this permission, make certain that decoding of authorization messages does not violate your organization's security policies. Cloudera Director should work without this permission if your IAM policy includes the required permissions specified above.

```
{
  "Statement": [
    {
      "Sid": "directorEc2",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeImages",
        "ec2:DescribeInstanceStatus",
        "ec2:DescribeInstances",
        "ec2:DescribeKeyPairs",
        "ec2:DescribePlacementGroups",
        "ec2:DescribeRegions",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:RunInstances",
        "ec2:TerminateInstances"
      ],
      "Resource": "*"
    },
    {
      "Sid": "directorIam",
      "Effect": "Allow",
      "Action": [
        "iam:GetInstanceProfile",
        "iam:PassRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "directorRds",
      "Effect": "Allow",
      "Action": [
        "rds:CreateDBInstance",
```

```

        "rds:DeleteDBInstance",
        "rds:DescribeDBInstances",
        "rds:DescribeDBSecurityGroups"
    ],
    "Resource": "*"
  },
  {
    "Sid": "directorSts",
    "Action": [
      "sts:DecodeAuthorizationMessage"
    ],
    "Effect": "Allow",
    "Resource": "*"
  }
]
}

```

Using MySQL for Cloudera Director Server



Note: This section is about the data Cloudera Director server stores for its own use. You can also use external databases for Cloudera Manager and cluster services. For more information, see [Using an External Database for Cloudera Manager and Clusters](#) on page 57.

Cloudera Director stores various kinds of data, including information about deployments, database servers, users, CDH clusters, and Cloudera Manager instances. By default, this data is stored in an embedded H2 database stored on the filesystem where the server is running at the following location:

```

/var/lib/cloudera-director-server/state.h2.db

```

Alternatively, you can use a MySQL database instead of the embedded H2 database, as described below.

Installing the MySQL Server



Note:

- If you already have a MySQL database set up, you can skip to [Configuring and Starting the MySQL Server](#) on page 51 to verify that your MySQL configuration meets the requirements for Cloudera Director.
- The `datadir` directory (`/var/lib/mysql` by default) must be located on a partition that has sufficient free space.

1. Install the MySQL database.

OS	Command
RHEL	\$ sudo yum install mysql-server
SLES	\$ sudo zypper install mysql \$ sudo zypper install libmysqlclient_r15
Ubuntu and Debian	\$ sudo apt-get install mysql-server

Note: Some SLES systems encounter errors with the `zypper install` command. For more information, see the Novell Knowledgebase topic, [error running chkconfig](#).

After issuing the command, you may need to confirm that you want to complete the installation.

Configuring and Starting the MySQL Server

1. Determine the version of MySQL.
2. Stop the MySQL server if it is running.

OS	Command
RHEL	\$ sudo service mysqld stop
SLES, Ubuntu, and Debian	\$ sudo service mysql stop

3. Move old InnoDB log files `/var/lib/mysql/ib_logfile0` and `/var/lib/mysql/ib_logfile1` from `/var/lib/mysql/` to a backup location.

4. Determine the location of the [option file](#), `my.cnf`, and update it as follows::

- To prevent deadlocks, set the isolation level to read committed.
- Configure MySQL to use the InnoDB engine, rather than MyISAM. (The default storage engine for MySQL is MyISAM.) To check which engine your tables are using, run the following command from the MySQL shell:

```
mysql> show table status;
```

- To configure MySQL to use the InnoDB storage engine, add the following line to the `[mysqld]` section of the `my.cnf` option file:

```
[mysqld]
default-storage-engine = innodb
```

- Binary logging is not a requirement for Cloudera Director installations. Binary logging provides benefits such as MySQL replication or point-in-time incremental recovery after database restore. Examples of this configuration follow. For more information, see [The Binary Log](#).

Following is a typical option file:

```
[mysqld]
default-storage-engine = innodb
transaction-isolation = READ-COMMITTED
# Disabling symbolic-links is recommended to prevent assorted security risks;
# to do so, uncomment this line:
# symbolic-links = 0

key_buffer = 16M
key_buffer_size = 32M
max_allowed_packet = 32M
thread_stack = 256K
thread_cache_size = 64
query_cache_limit = 8M
query_cache_size = 64M
query_cache_type = 1

max_connections = 550

#log_bin should be on a disk with enough free space. Replace
'/var/lib/mysql/mysql_binary_log' with an appropriate path for your system.
#log_bin=/var/lib/mysql/mysql_binary_log
#expire_logs_days = 10
#max_binlog_size = 100M

# For MySQL version 5.1.8 or later. Comment out binlog_format for older versions.
binlog_format = mixed


read_buffer_size = 2M
read_rnd_buffer_size = 16M
sort_buffer_size = 8M
join_buffer_size = 8M

# InnoDB settings
```

```
innodb_file_per_table = 1
innodb_flush_log_at_trx_commit = 2
innodb_log_buffer_size = 64M
innodb_buffer_pool_size = 4G
innodb_thread_concurrency = 8
innodb_flush_method = O_DIRECT
innodb_log_file_size = 512M

[mysqld_safe]
log-error=/var/log/mysql/mysql.log
pid-file=/var/run/mysql/mysql.pid
```

- If AppArmor is running on the host where MySQL is installed, you might need to configure AppArmor to allow MySQL to write to the binary.
- Ensure that the MySQL server starts at boot.

OS	Command
RHEL	<pre>\$ sudo /sbin/chkconfig mysqld on \$ sudo /sbin/chkconfig --list mysqld mysqld 0:off 1:off 2:on 3:on 4:on 5:on 6:off</pre>
SLES	<pre>\$ sudo chkconfig --add mysql</pre>
Ubuntu and Debian	<pre>\$ sudo chkconfig mysql on</pre> <div style="border: 1px solid black; padding: 10px; margin-top: 10px;"> <p> Note: <code>chkconfig</code> may not be available on recent Ubuntu releases. You may need to use Upstart to configure MySQL to start automatically when the system boots. For more information, see the Ubuntu documentation or the Upstart Cookbook.</p> </div>

- Start the MySQL server:


OS	Command
RHEL	<pre>\$ sudo service mysqld start</pre>
SLES, Ubuntu, and Debian	<pre>\$ sudo service mysql start</pre>

- Set the MySQL root password. In the following example, the current `root` password is blank. Press the **Enter** key when you're prompted for the root password.

```
$ sudo /usr/bin/mysql_secure_installation
[...]
Enter current password for root (enter for none):
OK, successfully used password, moving on...
[...]
Set root password? [Y/n] y
New password:
Re-enter new password:
Remove anonymous users? [Y/n] Y
[...]
Disallow root login remotely? [Y/n] N
[...]
Remove test database and access to it [Y/n] Y
[...]
Reload privilege tables now? [Y/n] Y
All done!
```

Installing the MySQL JDBC Driver

Install the MySQL JDBC driver for the Linux distribution you are using.

OS	Command
RHEL 5 or 6	<ol style="list-style-type: none"> Download the MySQL JDBC driver from the Download Connector/J page of the MySQL web site. Extract the JDBC driver JAR file from the downloaded file. For example: <pre>tar zxvf mysql-connector-java-version.tar.gz</pre> Add the JDBC driver, renamed, to the relevant server. For example: <pre>\$ sudo cp mysql-connector-java-version/mysql-connector-java-version-bin.jar /usr/share/java/mysql-connector-java.jar</pre> <p>If the target directory does not yet exist on this host, you can create it before copying the JAR file. For example:</p> <pre>\$ sudo mkdir -p /usr/share/java/ \$ sudo cp mysql-connector-java-version/mysql-connector-java-version-bin.jar /usr/share/java/mysql-connector-java.jar</pre> <div style="border: 1px solid green; padding: 5px; margin-top: 10px;">  Note: Do not use the <code>yum install</code> command to install the MySQL connector package, because it installs the openJDK, and then uses the Linux <code>alternatives</code> command to set the system JDK to be the openJDK. </div>
SLES	<pre>\$ sudo zypper install mysql-connector-java</pre>
Ubuntu or Debian	<pre>\$ sudo apt-get install libmysql-java</pre>

Creating a Database for Cloudera Director Server

You can create the database on the host where the Cloudera Director server will run, or on another host that is accessible by the Cloudera Director server. The database must be configured to support UTF-8 character set encoding.

Record the values you enter for database names, user names, and passwords. Cloudera Director requires this information to connect to the database.

- Log into MySQL as the root user:

```
$ mysql -u root -p
Enter password:
```

- Create a database for Cloudera Director server:

```
mysql> create database database DEFAULT CHARACTER SET utf8;
Query OK, 1 row affected (0.00 sec)

mysql > grant all on database.* TO 'user'@'%' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.00 sec)
```

database, *user*, and *password* can be any value. The examples match the names you provide in the Cloudera Director configuration settings described below in [Configure Cloudera Director Server to use the MySQL Database](#).

Backing Up MySQL Databases

To back up the MySQL database, run the `mysqldump` command on the MySQL host, as follows:

```
$ mysqldump -hhostname -uusername -ppassword database > /tmp/database-backup.sql
```

Configuring Cloudera Director Server to use the MySQL Database

Before starting the Cloudera Director server, edit the "Configurations for database connectivity" section of `/etc/cloudera-director-server/application.properties`.



Note: If the Cloudera Director server is already running, it must be restarted after configuring MySQL access. The server will not load configuration updates while running.

```
#
# Configurations for database connectivity.
#
# Optional database type (h2 or mysql) (defaults to h2)
#lp.database.type: mysql
#
# Optional database user name (defaults to "director")
#lp.database.username:
#
# Optional database password (defaults to "password")
#lp.database.password:
#
# Optional database host (defaults to "localhost")
#lp.database.host:
#
# Optional database port (defaults to 3306)
#lp.database.port:
#
# Optional database (schema) name (defaults to "director")
#lp.database.name:
```

Cloudera Director Database Encryption

The Cloudera Director server stores sensitive data in its database, including SSH credentials and cloud provider keys. You can configure Cloudera Director to encrypt the data stored in the Cloudera Director database.



Note: This section discusses data stored in the Cloudera Director database, not data stored in databases used by Cloudera Manager or cluster services.

Cipher Configuration

Database encryption is configured by setting the two server configuration properties described in the following table.

Table 1: Server Configuration Properties

Property	Description
<code>lp.encryption.twoWayCipher</code>	Cipher used to encrypt data. Possible values: <ul style="list-style-type: none"> <code>desede</code> - Triple DES (default) <code>passthrough</code> - No encryption <code>transitional</code> - Changing encryption
<code>lp.encryption.twoWayCipherConfig</code>	The configuration string for the chosen cipher.

The format of the configuration string varies with the choice of cipher, as described in the table below:

Table 2: Ciphers and Configuration Strings

Cipher	Configuration String Format
desede	24-byte symmetric encryption key, encoded as a string using Base64
passthrough	ignored
transitional	combination of old cipher and new cipher (see below)

The default value for the configuration string is a fixed 24-byte key for the default triple DES encryption:

```
ZGVmYXVsdGRpcmVjdG9yZGVzZWR1a2V5
```



Important: Cloudera highly recommends that you configure a different triple DES key. A warning appears in the server log if the default key is detected.

Starting with Encryption

Cloudera Director's default configuration for database encryption encrypts new data stored in the Cloudera Director database. This default configuration uses triple DES encryption, with a default key, to protect data. In a new installation of Cloudera Director, all data needing protection will be encrypted under the default encryption scheme. In an installation that was previously not configured for encryption, including older releases of Cloudera Director, new data needing protection will be encrypted, but old data needing protection will remain unencrypted until it is updated in the database over time.

If this level of protection is sufficient for your needs, it is not necessary to make any changes to Cloudera Director configuration. While Cloudera Director will function correctly, keep in mind that there are drawbacks: some data needing protection in the database may remain unencrypted indefinitely, and data that is encrypted is effectively only obscured, since the default key is not secret.

Establishing More Secure Encryption for New Installations

For a new installation of Cloudera Director, it is recommended that you generate and configure your own secret encryption key, different from the default key. Create a new key by generating 24 bytes of random data from a cryptographically secure random generator, and encode the bytes using the Base64 encoding algorithm.

Here is an example of generating a new key using Python.

```
python -c 'import base64, os; print base64.b64encode(os.urandom(24))'
```

Set the Cloudera Director configuration property `lp.encryption.twoWayCipherConfig` to the Base64-encoded key string before starting Cloudera Director for the first time. All data needing protection in the database will be encrypted with this key. It is good practice to change the encryption key periodically to protect against unintentional disclosure. See [Changing Encryption](#) below for more.



Note: If you configure a new secret key, Cloudera recommends you restrict permissions on the configuration file (`application.properties`) to protect the key from disclosure. Ensure that at least the user running Cloudera Director can still read the file.

Establishing More Secure Encryption for Existing Installations

For an existing installation of Cloudera Director that uses either no encryption at all (including older releases of Cloudera Director) or uses only the default encryption, it is recommended that you use a transitional cipher to change encryption to a more secure state. Not only will changing encryption introduce the use of a non-default and secret key, but it will also forcibly encrypt all data needing protection in the database, whether it was already encrypted or not.

See [Changing Encryption](#) below for details on how to configure a transitional cipher in order to change encryption. When configuring the transitional cipher, you will need to know information about the old cipher that was in effect.

- If the default cipher and key was in use previously, then use "desede" and the default key for the old cipher configuration.
- If no encryption was in place previously, including older releases of Cloudera Director which did not support database encryption, then use "passthrough" (with no configuration string) for the old cipher configuration.

The new cipher should be triple DES ("desede") with a secret key that you generate. See [Establishing More Secure Encryption for New Installations](#) above for details on how to generate a good key.

After establishing more secure encryption, it is good practice to change the encryption key periodically to protect against unintentional disclosure. Use the transitional cipher again to change encryption to use a new key.

Changing Encryption

To change the key used for database encryption, or change to a different cipher, you must configure the Cloudera Director server to use a transitional cipher.



Note: Transitional ciphers are supported for Cloudera Director server only, not for Cloudera Director client.

If a transitional cipher is configured, Cloudera Director encrypts all data that needs protection, changing from an old encryption scheme to a new encryption scheme. A transitional cipher can change the encryption in effect, or introduce it when it has not been used before, including under older Cloudera Director releases. It also ensures that all data needing protection becomes encrypted.

To configure a transitional cipher:

1. Stop the server.
2. Configure `lp.encryption.twoWayCipher` with the value `transitional`.
3. Configure `lp.encryption.twoWayCipherConfig` with a configuration string describing both the old cipher and the new cipher.
4. Start the server.

The configuration string for a transitional cipher has the following format:

```
old-cipher;old-configuration-string|new-cipher;new-configuration-string
```

For example, to change the triple DES key, use a configuration string like this:

```
desede;old-key-in-base64|desede;new-key-in-base64
```

To transition from the default triple DES encryption key to a new key, use a configuration string like this:

```
desede;ZGVmYXVsdGRpcmVjdG9yZGVzZWRLa2V5|desede;new-key-in-base64
```

To transition from no encryption to triple DES encryption with a new key, use a configuration string like this:

```
passthrough;|desede;new-key-in-base64
```

A transitional cipher cannot be used as the old or new cipher in another transitional cipher.

When the server restarts, it detects that a transitional cipher is configured and updates all relevant data, unencrypted and encrypted, to the new cipher. After this process is complete, the server continues startup as usual. Configuring a transitional cipher ensures that all data needing protection in the database is encrypted.

There is an important restriction on changing encryption. If any work is waiting to be resumed by the server on startup (for example, bootstrapping a new cluster), then the server will refuse to change encryption and will stop. You must configure the server for its old cipher, start it, and wait for that work to resume and be completed.

After encryption has been changed using a transitional cipher, you can stop the server again and configure it to use the new cipher normally. The server can continue to run configured with a transitional cipher, but it should be reconfigured to use a normal cipher before its next restart, or else the server will not restart if work is waiting to be resumed.

Using an External Database for Cloudera Manager and Clusters

By default, Cloudera Director configures Cloudera Manager and cluster services, such as Hive, to use the Cloudera Manager embedded PostgreSQL database. You can also configure Cloudera Director to use external database servers, instead. If you have a database server configured, you can use Cloudera Director to configure Cloudera Manager and cluster services to create or use databases on that server. You can also configure Cloudera Director to use a cloud provider, like Amazon's Relational Database Service (RDS), to provision new database servers.

How you set up external database servers and databases differs depending on whether you are using Cloudera Director client or Cloudera Director server:

- **Cloudera Director client** - Configure external databases in the `cluster.conf` file and launch Cloudera Director client (standalone) by issuing the `bootstrap` command.
- **Cloudera Director server** - Configure external databases with the API. You can also configure external databases through the UI or by editing the `cluster.conf` file and launching Cloudera Director server with the `bootstrap-remote` command.

Draft Comment:

Add info (to second bullet point above) on configuring external dbs for Director server using the UI.

These procedures are described in the topics in this section.

Defining Database Servers

Cloudera Director must have information about external database servers before it can use them. A single database server is scoped to an environment, so only deployments and clusters in that environment recognize it.

A database server template can refer to either an existing server or a database server to be created. Following are the basic elements of a database server template.

- **name** - A unique name for the server within the environment
- **type** - The type of database server, such as "MYSQL" or "POSTGRESQL"
- **hostname** - The name of the server host
- **port** - The listening port of the server
- **username** - The name of the administrative account for the server
- **password** - The password for the administrative account

The hostname and port are optional in a template. If they are not present, then Cloudera Director assumes that the template refers to a server that does not yet exist and must be created.

A database server template also supports a table of key-value pairs of configuration information, which Cloudera Director may require when creating a new server. A template also supports a second table of tag data, which Cloudera Director can employ for certain cloud providers, including Amazon Web Services.

API

The Cloudera Director server has a REST service endpoint for managing external database server definitions. The operations supported by the endpoint are described in the table below.

- Each service URI begins with `"/api/v2/environments/{environment}"`, where `"{environment}"` is the name of the environment within which the database server definition is scoped.
- They all use JSON for input data and response data.

Operation	Description	Notes
POST /databaseServers/	Define a new database.	Admin required.
GET /databaseServers/	List all database servers.	
DELETE /databaseServers/{name}	Delete a database server definition.	Admin required.
PUT /databaseServers/{name}	Update a database server definition.	Admin required.
GET /databaseServers/{name}	Get a database server definition.	
GET /databaseServers/{name}/status	Get the status of a database server.	
GET /databaseServers/{name}/template	Get the template from which a database server was defined.	

If a database server template without a host and port is posted to Cloudera Director, Cloudera Director will asynchronously begin the process of creating the server on a cloud provider. The provider is selected based on the environment.

Similarly, if a database server definition is deleted, and the server was originally created by Cloudera Director, Cloudera Director will begin the process of deleting the database from the cloud provider. Before deleting a server definition, be sure to make any backups of the server that you need.

The status of a database server indicates its current position in the server lifecycle. The following values can be returned by the GET database server status operation:

Status	Description
BOOTSTRAPPING	Cloudera Director is in the process of creating the server.
BOOTSTRAP_FAILED	Cloudera Director failed to create the server.
READY	The server is available for use.
TERMINATING	Cloudera Director is in the process of destroying the server.
TERMINATE_FAILED	Cloudera Director failed to terminate the server.
TERMINATED	The server has been destroyed.

Client Configuration File (databaseServers section)

Database server templates can be provided in the configuration file passed to the Cloudera Director standalone client. Define external database servers in the `databaseServers` section of a configuration file.

See the API section above for a description of the different parts of a template. The following example defines two existing database servers.

```

databaseServers {
  mysql {
    type: mysql
    host: 1.2.3.4
    port: 3306
    user: root
    password: password
  }
  postgres1 {
    type: postgresql
    host: 1.2.3.4
    port: 5432
    user: postgres
    password: password
  }
}

```

The following example defines a server that Cloudera Director must create using RDS.

```
databaseServers {
  mysqlt1 {
    type: mysql
    user: root
    password: password
    instanceClass: db.m3.medium
    engineVersion: 5.5.40b
    dbSubnetGroupName: default
    vpcSecurityGroupIds: sg-abcd1234
    allocatedStorage: 10
    tags {
      owner: jsmith
    }
  }
}
```

You cannot include both existing servers, and servers that Cloudera Director must create, in the same configuration file. You can create new database servers separately in a cloud provider and then define them as existing servers in the configuration file.

Using Amazon RDS for External Databases

Cloudera Director can use Amazon Relational Database Service (RDS) to create new database servers. These servers can be used to host external databases for Cloudera Manager and CDH cluster services.



Note:

- At this time, only MySQL 5.5 and 5.6 RDS instances are supported.
- RDS works through both `bootstrap-remote` and standalone `bootstrap` on the client, as well as through the server API. It is not supported through the UI.

Creating a Template to Use Amazon RDS as an External Database

An external database server to be created on RDS is defined by a template just like any other server, except that the host and port are not specified; these are determined as the server is being created.

- **name** - A unique name for the server within the environment
- **type** - The type of database server, such as "MYSQL"
- **username** - The name of the administrative account for the server
- **password** - The password for the administrative account

The key-value configuration information in the template for an RDS server must include information required by RDS to create a new instance. Cloudera recommends that you specify the engine version in a template. If you do not specify the version, RDS defaults to a recent version, which can change over time.



Note: If you are including Hive in your clusters, and you configure the Hive metastore to be installed on MySQL through RDS, Cloudera Manager may report that "The Hive Metastore canary failed to create a database." This is caused by a MySQL bug that is exposed through using MySQL 5.6.5 or later with the MySQL JDBC driver (used by Cloudera Director) version 5.1.19 or earlier. Cloudera recommends that you use a MySQL version that avoids revealing this bug for the driver version installed by Cloudera Director from your platform software repositories.

key	description	example
instanceClass	Instance type for database server instance	db.m3.medium
dbSubnetGroupName	Name of the DB subnet group which the instance spans	default

key	description	example
engineVersion	(optional) Version of database engine	5.5.40b
vpcSecurityGroupIds	Comma-separated list of security groups for the new instance	sg-abc123,sg-def456
allocatedStorage	Storage in gigabytes for new server	10
availabilityZone	(optional) Preferred availability zone for the new server	us-east-1d



Note:

- Cloudera Director does not currently support creating multi-AZ instances.
- The template can also specify tags for the new instance.

Defining a Database Server in AWS Using RDS: API

Use the previously described [REST service endpoint](#) for external database server definitions to create and destroy external database servers using RDS. The environment in which servers are defined must already be configured to use AWS, and your account must have permission to create and delete RDS instances.

When an external database server template is submitted via POST to the endpoint, and the template lacks a host and port, Cloudera Director accepts the definition for the server and asynchronously begins the process of creating the new server. The complete existing server definition, including the host and port, will eventually be available via GET.

Likewise, when the definition is deleted via DELETE, Cloudera Director begins destroying the server.

While a new server is being created on RDS, you may begin the process of bootstrapping new deployments and new clusters whose external database templates refer to the server. The bootstrap process will proceed in tandem with the server creation, and pause when necessary to wait for the new RDS instance to be available for use.

When a deployment or cluster is terminated, Cloudera Director leaves RDS instances alone. This makes it possible for multiple deployments and clusters to share the same external database servers that Cloudera Director creates on RDS.

Defining a Database Server Using RDS: Client Configuration File

The following example defines a server that Cloudera Director must create using RDS:

```
databaseServers {
  mysqlt1 {
    type: mysql
    user: root
    password: password
    instanceClass: db.m3.medium
    engineVersion: 5.5.40b
    dbSubnetGroupName: default
    vpcSecurityGroupIds: sg-abcd1234
    allocatedStorage: 10
    tags {
      owner: jsmith
    }
  }
}
```

The following example of an external database template uses the new server that Cloudera Director needs to create. The databaseServerName item matches the name of the new server:

```
cluster {
  #... databaseTemplates: {
  HIVE {
    name: hivetemplate
    databaseServerName: mysqlt1
  }
}
```

```

    databaseNamePrefix: hivemetastore
    usernamePrefix: hive
  }
}

```

Using External Databases

After external database servers are defined, the databases on them can be defined. Cloudera Director can use databases that already exist on those servers, or it can create them while bootstrapping new Cloudera Manager installations or CDH clusters.

The following parts of an existing database must be defined:

- **type** - The type of database, “MYSQL” or “POSTGRESQL.”
- **hostname** - The name of the server host.
- **port** - The listening port of the server.
- **name** - The name of the database on the server.
- **username** - The name of the user account having full access to the database.
- **password** - The password for the user account.

The parts of an external database template are:

- **name** - A unique name for the template within the deployment or cluster template.
- **databaseServerName** - The name of the external database server where the new database is to reside.
- **databaseNamePrefix** - The string prefix for the name of the new database server.
- **usernamePrefix** - The string prefix for the name of the new user account that will have full access to the database.

The database server name in a database server template must refer to an external database server that is already defined.

When Cloudera Director creates the new database, it names the database by starting with the prefix in the template and then appends a random string. This prevents name duplication issues when sharing a database server across many deployments and clusters. Likewise, Cloudera Director creates new user accounts by starting with the prefix in the template and appending a random string.



Important: If you are using a MySQL database, the `usernamePrefix` you define should be no more than seven characters long. This keeps usernames generated by Cloudera Director within the MySQL limit of sixteen characters for usernames.

If Cloudera Director creates new external databases during the bootstrap of a deployment or cluster, then it also drops them, and their associated user accounts, when terminating the deployment or cluster. Be sure to back up those databases before beginning termination.



Note: Cloudera Director cannot create databases on remote database servers that Cloudera Director (or code that it runs) is unable to reach. For example, Cloudera Director cannot work with a database server that only allows local access, unless that server happens to be on the same machine as Cloudera Director. Use the following workarounds:

- Reconfigure the database server, and any security measures that apply to it, to allow Cloudera Director access during the bootstrap and termination processes.
- Open an SSH tunnel for database server access.
- Create the databases manually and configure them using normal Cloudera Director support for external databases.

API

Define external databases in the templates for new Cloudera Manager installations (“deployments”) or new clusters. You cannot define both existing databases, and new databases that need to be created, in the same template.

Defining External Databases in the Configuration File

For Cloudera Manager

Define external databases used by Cloudera Manager in the `cloudera-manager` section of a configuration file. The following example defines existing external databases.

```
cloudera-manager {
  # ...
  databases {
    CLOUDERA_MANAGER {
      name: scm1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: scmuser
      password: scmpassword
    }
    ACTIVITYMONITOR {
      name: aml
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: amuser
      password: ampassword
    }
    REPORTSMANAGER {
      name: rml
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: rmuser
      password: rmpassword
    }
    NAVIGATOR {
      name: nav1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: navuser
      password: navpassword
    }
    NAVIGATORMETASERVER {
      name: navmetal
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: navmetauser
      password: navmetapassword
    }
  }
}
```

The following example defines new external databases that Cloudera Director must create while bootstrapping the deployment.

```
cloudera-manager {
  # ...
  databaseTemplates {
    CLOUDERA_MANAGER {
      name: cmtemplate
      databaseServerName: mysql1
      databaseNamePrefix: scm
      usernamePrefix: cmadmin
    }
    ACTIVITYMONITOR {
```

```

    name: cmamtemplate
    databaseServerName: mysql1
    databaseNamePrefix: am
    usernamePrefix: cmamadmin
  }
  REPORTSMANAGER {
    name: cmrmtemplate
    databaseServerName: mysql1
    databaseNamePrefix: rm
    usernamePrefix: cmradmin
  }
  NAVIGATOR {
    name: cmnavtemplate
    databaseServerName: mysql1
    databaseNamePrefix: nav
    user: cmnavadmin
  }
  NAVIGATORMETASERVER {
    name: cmnavmetatemplate
    databaseServerName: mysql1
    databaseNamePrefix: navmeta
    usernamePrefix: cmnavmetaadmin
  }
}

```

Each template must refer to a database server defined elsewhere in the configuration file. The database server template can be for a server that does not yet exist; in that case, Cloudera Director starts creating the server, and then waits while bootstrapping the deployment until the server is available.

A deployment must use either all existing databases or all non-existing databases for the different Cloudera Manager components; they cannot be mixed.

For Cluster Services

Define external databases used by cluster services such as Hive in the `cluster` section of a configuration file. The following example defines existing external databases.

```

cluster {
  #...
  databaseTemplates: {
    HIVE {
      name: hive1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: hiveuser
      password: hivepassword
    }
  }
}

```

The following example defines new external databases that Cloudera Director must create while bootstrapping the cluster.

```

cluster {
  #...
  databaseTemplates: {
    HIVE {
      name: hivetemplate
      databaseServerName: mysql1
      databaseNamePrefix: hivemetastore
      usernamePrefix: hive
    }
  }
}

```

Each template must refer to a database server defined elsewhere in the configuration file. The database server template can be for a server that does not yet exist; in that case, Cloudera Director starts creating the server, and then waits while bootstrapping the cluster until the server is available.

A deployment must use either all existing databases or all non-existing databases for the different cluster services; they cannot be mixed.

Setting Cloudera Director Properties

This topic lists the configuration properties recognized by Cloudera Director. Upon installation, these properties are pre-configured with reasonable default values, and you can run either client or server versions without specifying any of them. However, you might want to customize one or more properties, depending on your environment and the Cloudera Director features you want to use.

Setting Configuration Properties

The Cloudera Director command line provides the simplest way to specify a configuration property. For example:

```
./bin/cloudera-director bootstrap aws.simple.conf \  
--lp.pipeline.retry.maxWaitBetweenAttempts=60
```

```
./bin/cloudera-director-server --lp.security.disabled=false
```

Tip: If you want to configure many properties, add them to the `etc/application.properties` file in the Cloudera Director installation. The properties in this file take effect automatically. To override these properties, set new values in the command line.

For users upgrading to Cloudera Director 1.5 from Cloudera Director 1.1.x

If you modified the `application.properties` file in Cloudera Director 1.1.x, the result of an upgrade to Cloudera Director 1.5 depends on the version of Linux you are using:

- **RHEL and CentOS** - The `application.properties` file will not be overwritten with the new 1.5 version properties when you upgrade. Instead, new properties introduced in Cloudera Director 1.5 will be added to `application.properties.rpmnew`.
- **Debian and Ubuntu** - The modified Cloudera Director 1.1.x `application.properties` file will be backed up and then overwritten by the new `application.properties` file containing new Cloudera Director 1.5 properties.

All the new 1.5 properties are commented, and they all use valid defaults, so you do not necessarily need to merge the two properties files. But you must merge the two files if you want to modify one of the newly introduced properties.

Property Types

Type	Description
boolean	Either true or false
char	Single character
directory	Valid directory path
enum	Fixed set of string values; a list of each enumeration's values is provided following the main property table below
enum list	Comma-separated list of enums
file	Valid file path
int	Integer (32-bit)
long	Long integer (64-bit)
string	Ordinary character string

Type	Description
time unit	Enumeration of time units: DAYS, HOURS, MICROSECONDS, MILLISECONDS, MINUTES, NANOSECONDS, SECONDS

Properties

Property	Description
<code>lp.access.logging.config.file</code>	File for Cludera Director server access log. Type: string Default: none; must be set if <code>lp.access.logging.enabled</code> is true.
<code>lp.access.logging.enabled</code>	Enable Cludera Director server access logging. Type: boolean Default: false
<code>lp.bootstrap.agents.maxNumberOfInstallAttempts</code>	Maximum number of times to retry installing Cludera Manager agent. Use -1 for unlimited. Type: int Default: -1
<code>lp.bootstrap.parallelBatchSize</code>	Parallelism for allocating and setting up cluster instances when bootstrapping a cluster. Type: int Default: 20
<code>lp.bootstrap.parcels.distributeMaxConcurrentUploads</code>	Maximum concurrent uploads of parcels across cluster. Type: int Default: 5
<code>lp.bootstrap.parcels.distributeRateLimitKbs</code>	Maximum rate of parcel upload, in KB/s. Type: int Default: 256000
<code>lp.bootstrap.resume.policy</code>	Action to take when resuming a previous bootstrap. Use RESTART to start from scratch. Use RESUME to resume from last known state. Use INTERACTIVE to prompt to ask. Type: enum Valid values: RESTART RESUME INTERACTIVE Default: INTERACTIVE
<code>lp.cache.health.expirationMultiplier</code>	Multiplier applied to polling rate to find health cache expiration duration; negative = disable health polling. Type: int Default: 2

Property	Description
<code>lp.cache.health.numberOfCacheExecutionThreads</code>	Number of threads used to poll for service and cluster health. Type: int Default: 5
<code>lp.cache.health.pollingRateInMilliseconds</code>	Rate at which service and cluster health is polled, in milliseconds. Type: long Default: 30000
<code>lp.cleanup.databases.intervalBetweenAttemptsInMs</code>	Wait time between attempts to destroy external databases, in milliseconds. Type: long Default: 60000
<code>lp.cleanup.databases.maxNumberOfDeleteAttempts</code>	Maximum number of times to retry destroying external databases; -1 = unlimited. Type: int Default: 5
<code>lp.cloud.databaseServers.allocate.timeoutInMinutes</code>	Time to wait for allocated database server instances to begin running to have ports available. Type: int Default: 20
<code>lp.cloud.databaseServers.destroy.timeoutInMinutes</code>	Time to wait for terminated database server instances to stop running to have ports no longer available. Type: int Default: 20
<code>lp.cloud.instances.allocate.numberOfRetriesOnConnectionError</code>	Number of times to retry connecting to newly allocated instances over SSH. Type: int Default: 3
<code>lp.cloud.instances.allocate.parallelBatchSize</code>	Parallelism for waiting for SSH to become available on newly allocated instances. Type: int Default: 20
<code>lp.cloud.instances.allocate.timeBetweenConnectionRetriesInSeconds</code>	Time to wait between attempts to connect to newly allocated instances over SSH. Type: int Default: 1

Property	Description
<code>lp.cloud.instances.allocate.timeoutInMinutes</code>	Time to wait for allocated instances to begin running to have SSH ports available. Type: int Default: 20
<code>lp.cloud.instances.terminate.timeoutInMinutes</code>	Time to wait for terminated instances to stop running. Type: int Default: 20
<code>lp.debug.collectDiagnosticDataOnFailure</code>	Collect Cloudera Manager diagnostic data on unrecoverable bootstrap failure. Type: boolean Default: true
<code>lp.debug.createDiagnosticDataDownloadDirectory</code>	Create the download directory for Cloudera Manager diagnostic data if it does not already exist. Type: boolean Default: true
<code>lp.debug.diagnosticDataDownloadDirectory</code>	Destination directory for downloaded Cloudera Manager diagnostic data. Type: string Default: /tmp
<code>lp.debug.downloadDiagnosticData</code>	Download Cloudera Manager diagnostic data once it has been collected. Type: boolean Default: true
<code>lp.debug.dumpClouderaManagerLogsOnFailure</code>	Dump Cloudera Manager log entries into the Director logs on unrecoverable bootstrap failure. Type: boolean Default: false
<code>lp.debug.dumpClusterLogsOnFailure</code>	Dump cluster service logs, standard output, or standard error into the Cloudera Director logs on unrecoverable bootstrap failure. Type: boolean Default: false
<code>lp.encryption.twoWayCipher</code>	Cipher used to encrypt data. Possible values: <ul style="list-style-type: none"> • <code>desede</code> - Triple DES • <code>passthrough</code> - No encryption • <code>transitional</code> - Changing encryption Type: string Default: <code>desede</code>

Property	Description
<code>lp.encryption.twoWayCipherConfig</code>	<p>The configuration string for the chosen cipher.</p> <p>Type: string</p> <p>Default: ZGVmYXVsdGRpcmVjdG9yZGVzZWR1a2V5</p> <div style="border: 1px solid orange; padding: 5px; margin-top: 10px;"> <p> Important: Cloudera recommends that you configure a different triple DES key. A warning appears in the server log if the default key is detected.</p> </div>
<code>lp.metrics.durationUnits</code>	<p>Time units for reporting durations in metrics.</p> <p>Type: time unit</p> <p>Valid values: DAYS HOURS MICROSECONDS MILLISECONDS MINUTES NANOSECONDS SECONDS</p> <p>Default: MILLISECONDS</p>
<code>lp.metrics.enabled</code>	<p>Enable metrics gathering</p> <p>Type: boolean</p> <p>Default: false</p>
<code>lp.metrics.location</code>	<p>Directory for storing metrics reports.</p> <p>Type: directory</p> <p>Default: \$LOG_DIR/metrics</p>
<code>lp.metrics.rateUnits</code>	<p>Time units for reporting rates in metrics.</p> <p>Type: time unit</p> <p>Valid values: DAYS HOURS MICROSECONDS MILLISECONDS MINUTES NANOSECONDS SECONDS</p> <p>Default: SECONDS</p>
<code>lp.metrics.reportingRate</code>	<p>Frequency of metrics reporting, in minutes.</p> <p>Type: long</p> <p>Default: 1</p>
<code>lp.pipeline.retry.maxNumberOfAttempts</code>	<p>Maximum number of times to retry failed pipeline jobs; -1 = unlimited.</p> <p>Type: int</p> <p>Default: -1 for client, 16 for server</p>
<code>lp.pipeline.retry.maxWaitBetweenAttempts</code>	<p>Maximum wait time between pipeline retry attempts, in seconds.</p> <p>Type: int</p> <p>Default: 45</p>

Property	Description
<code>lp.proxy.http.domain</code>	NT domain for HTTP proxy authentication; none = no domain. Type: string Default: none
<code>lp.proxy.http.host</code>	HTTP proxy host; none = no proxy. Type: string Default: none
<code>lp.proxy.http.password</code>	HTTP proxy password; none = no password. Type: string Default: none
<code>lp.proxy.http.port</code>	HTTP proxy port; -1 = no proxy. Type: int Default: -1
<code>lp.proxy.http.preemptiveBasicProxyAuth</code>	Whether to preemptively authenticate to HTTP proxy. Type: boolean Default: false
<code>lp.proxy.http.username</code>	HTTP proxy username; none = no username. Type: string Default: none
<code>lp.proxy.http.workstation</code>	Originating workstation in NT domain for HTTP proxy authentication; none = no workstation. Type: string Default: none
<code>lp.remote.hostAndPort</code>	Host and port of remote Cloudera Director server. Type: string Default: localhost:7189
<code>lp.remote.password</code>	Remote Cloudera Director server password (client only). Type: string Default:
<code>lp.remote.username</code>	Remote Cloudera Director server username (client only). Type: string Default: none
<code>lp.remote.terminate.assumeYes</code>	Whether to skip prompting user to confirm termination for client terminate-remote command. Type: boolean

Property	Description
	Default: false
<code>lp.security.enabled</code>	Whether to enable Cloudera Director server security (server only). Type: boolean Default: true
<code>lp.security.userSource</code>	Source for user account information (server only). Type: enum Default: internal
<code>lp.ssh.connectTimeoutInSeconds</code>	SSH connection timeout. Type: int Default: 30
<code>lp.ssh.heartbeatIntervalInSeconds</code>	SSH heartbeat interval. Type: int Default: 45
<code>lp.ssh.readTimeoutInSeconds</code>	SSH read timeout. Type: int Default: 30
<code>lp.task.evictionRate</code>	Rate of execution of database eviction, in milliseconds. Type: long Default: 600000
<code>lp.terminate.assumeYes</code>	Whether to skip prompting user to confirm termination for client terminate command. Type: boolean Default: false
<code>lp.terminate.deployment.clouderaManagerServerStopWaitTimeInMs</code>	Time to wait for Cloudera Manager to stop when terminating a deployment, in milliseconds. Type: long Default: 300000
<code>lp.terminate.deployment.timeBetweenConnectionRetriesInMs</code>	Time to wait between checks for whether Cloudera Manager has been terminated. Type: int Default: 10000
<code>lp.update.parallelBatchSize</code>	Parallelism for allocating and setting up cluster instances when bootstrapping a cluster. Type: int Default: 20

Property	Description
<code>lp.update.redeployClientConfigs.numberOfRetries</code>	Maximum number of times to retry deploying Cloudera Manager client configurations; -1 = unlimited. Type: int Default: 5
<code>lp.update.redeployClientConfigs.sleepAfterFailureInSeconds</code>	Wait time between attempts to deploy Cloudera Manager client configurations, in seconds. Type: int Default: 10
<code>lp.update.restartCluster.numberOfRetries</code>	Maximum number of times to retry a Cloudera Manager rolling restart; -1 = unlimited. Type: int Default: 5
<code>lp.update.restartCluster.rollingRestartSlaveBatchSize</code>	Number of instances with Cloudera Manager slave roles to restart at a time. Type: int Default: 20
<code>lp.update.restartCluster.rollingRestartSlaveFailCountThreshold</code>	Threshold for number of slave host batches that are allowed to fail to restart before the entire command is considered failed (advanced use only). Type: int Default: 0
<code>lp.update.restartCluster.rollingRestartSleepSeconds</code>	Number of seconds to sleep between restarts of Cloudera Manager slave host batches. Type: int Default: 0
<code>lp.update.restartCluster.sleepAfterFailureInSeconds</code>	Wait time between attempts to perform a Cloudera Manager rolling restart, in seconds. Type: int Default: 10
<code>lp.validate.dumpTemplates</code>	Whether to output validated configuration data as JSON. Type: boolean Default: false
<code>lp.webapp.anonymousUsageDataAllowed</code>	Allow Cloudera Director to send anonymous usage information to help Cloudera improve the product. Type: boolean Default: true

Property	Description
<code>lp.webapp.documentationType</code>	Whether Cloudera Director opens the latest help from the Cloudera web site (online) or locally installed help (embedded). Type: enumerated string {ONLINE, EMBEDDED} Default: ONLINE
<code>port</code>	Cloudera Director server port (server only). Type: int Default: 7189
<code>server.sessionTimeout</code>	Cloudera Director server session timeout (server only). Type: int Default: 18000

Setting Cloudera Manager Configurations

You can use Cloudera Director to set configurations for the various Cloudera Manager entities that it deploys:

- Cloudera Manager
- Cloudera Management Service
- The various CDH components, such as HDFS, Hive, and HBase
- Role types, such as NameNode, ResourceManager, and Impala Daemon

This functionality is available for both Cloudera Director client and Cloudera Director server:

- **Client** - Using the configuration file.
Draft Comment:
For more information on the configuration file, see The Cloudera Director Configuration File. [Add this back after the topic is updated... or keep it out but add an appendix containing the current contents of the file (with comments), so the details only have to be maintained in one place.]
- **Server** - Using the Cloudera Director UI or APIs (Java, REST, or Python).
 - To use the REST API, you can submit JSON documents to the REST service endpoint, or access the API console at `http://director-server-hostname:7189/api-console`.
 - You can find information about the Cloudera Director Java and Python APIs on the [director-sdk GitHub page](#).
 - In the UI, you can specify custom values for Cloudera Manager configurations when adding an environment or creating a Cloudera Manager cluster.



Note: Cloudera Manager configuration properties are case-sensitive. To verify the correct way to specify Cloudera Manager configuration properties in Cloudera Director API calls and in the configuration name fields of the Cloudera Director UI, see [Cloudera Manager Configuration Properties](#) in the Cloudera Manager documentation. By expanding this heading, you see topics such as the following:

- [CDH 5.4.0 Properties](#)
- [Host Configuration Properties](#)
- [Cloudera Manager Server Properties](#)
- [Cloudera Management Service](#)

These pages include tables of configuration properties. Locate the property whose value you want to customize, and use the name in the column **API Name**.

Cloudera Director enables you to customize deployment and cluster setup, and configurations are applied on top of Cloudera Manager default and automatic host-based configuration of services and roles. Set configurations either in the deployment template or in the cluster template.



Important: You can add Llama with autoconfiguration only if you are using Cloudera Manager 5.2 or later. For Cloudera Manager 5.0 and 5.1, you must set Llama configurations manually, as described in the [Cloudera Manager documentation](#).

Setting up a Cloudera Manager License

There are three ways to set up a Cloudera Manager license using Cloudera Director, each corresponding to a field within the `Licensing` configuration section of the `aws.conf` configuration file. The three are mutually exclusive.

- **license field** - You can embed license text in the `license` field of the configuration file. (Cloudera recommends using triple quotes ("") for including multi-line text strings, as shown in the commented-out lines of the configuration file.) To embed a license in the `license` field, find the `Licensing` configuration section of the configuration file and enter the appropriate values.
- **licensePath field** - The `licensePath` field can be used to specify the path to a file containing the license.
- **enableEnterpriseTrial field** - The `enableEnterpriseTrial` flag indicates whether the 60-Day Cloudera Enterprise Trial should be activated when no license is present. This must *not* be set to `true` if a license is included using either `license` or `licensePath`.

The `License` configuration section of the configuration file is shown below:

```
#
# Embed a license for Cloudera Manager
#
# license: ""
# -----BEGIN PGP SIGNED MESSAGE-----
# Hash: SHA1
#
# {
# "version" : 1,
# "name" : "License Owner",
# "uuid" : "license id",
# "expirationDate" : 0,
# "features" : [ "FEATURE1", "FEATURE2" ]
# }
# -----BEGIN PGP SIGNATURE-----
# Version: GnuPG v1.4.11 (GNU/Linux)
#
# PGP SIGNATURE
# -----END PGP SIGNATURE-----
# ""
#
# Include a license for Cloudera Manager from an external file
#
# licensePath: "/path/to/license.txt.asc"
#
# Activate 60-Day Cloudera Enterprise Trial
#
enableEnterpriseTrial: true
```

For more information about Cloudera Manager licenses, see [Managing Licenses](#) in the Cloudera Manager documentation.

Deployment Template Configuration

This section shows the structure of the Cloudera Manager deployment configuration settings in both the CLI and the API.

CLI

Using the CLI, the `configs` section in the deployment template has the following structure:

```
cloudera-manager {
  ...
  configs {
    # CLOUDERA_MANAGER corresponds to the Cloudera Manager Server configuration options

    CLOUDERA_MANAGER {
      enable_api_debug: false
    }

    # CLOUDERA_MANAGEMENT_SERVICE corresponds to the Service-Wide configuration options

    CLOUDERA_MANAGEMENT_SERVICE {
      enable_alerts : false
      enable_config_alerts : false
    }

    ACTIVITYMONITOR { ... }

    REPORTSMANAGER { ... }

    NAVIGATOR { ... }

    # Added in Cloudera Manager 5.2+
    NAVIGATORMETASERVER { ... }

    # Configuration properties for all hosts
    HOSTS { ... }
  }
  ...
}
```

API

Using the API, the `configs` section for deployment templates has the following structure:

```
{
  "configs": {
    "CLOUDERA_MANAGER": {
      "enable_api_debug": "true"
    },
    "CLOUDERA_MANAGEMENT_SERVICE": {
      "enable_alerts": "false"
    }
  }
}
```

Cluster Template Service-wide Configuration

This section shows the structure of the Cloudera Manager service-wide configuration settings in both the CLI and the API.

CLI

Using the CLI, the `configs` section for service-wide configurations in the cluster template has the following structure:

```
cluster {
  ...
  configs {
    HDFS {
      dfs_block_size: 1342177280
    }
    MAPREDUCE {
      mapred_system_dir: /user/home
      mr_user_to_impersonate: mapred1
    }
  }
}
```

```

    }
    ...
}

```

API

Using the API, the service-wide configurations block in the `ClusterTemplate` is labelled `servicesConfigs`, and has the following structure:

```

{
  "servicesConfigs": {
    "HDFS": {
      "dfs_block_size": 1342177280
    },
    "MAPREDUCE": {
      "mapred_system_dir": "/user/home",
      "mr_user_to_impersonate": "mapred1"
    }
  }
}

```

Cluster Template Roletype Configurations

This section shows the structure of the Cloudera Manager roletype configuration settings in both the CLI and the API.

CLI

Using the CLI, roletype configurations in the cluster template are specified per instance group:

```

cluster {
  ...
  masters {
    ...
    # Optional custom role configurations
    configs {
      HDFS {
        NAMENODE {
          dfs_name_dir_list: /data/nn
          namenode_port: 1234
        }
      }
    }
  }
  ...
}

```

API

Using the API, roletype configurations in the cluster template are specified per instance group:

```

{
  "virtualInstanceGroups" : {
    "configs": {
      "HDFS": {
        "NAMENODE": {
          "dfs_name_dir_list": "/data/nn",
          "namenode_port": "1234"
        }
      }
    }
  }
}

```

Configuring Cloudera Director for a New AWS Instance Type

Amazon Web Services occasionally introduces new instance types with improved specifications. Cloudera Director ships with the functionality needed to support all of the instance types available at the time of release, but customers can augment that to allow it to support new types that are introduced after release.

Updated Virtualization Mappings

Each Linux Amazon Machine Image (AMI) uses one of two types of virtualization, paravirtual or HVM. Cloudera Director ensures that the instance type of an instance that is to host an AMI supports the AMI's virtualization type. The knowledge of which instance types support which virtualizations resides in a virtualization mappings file.

The AWS plugin included with Cloudera Director ships with an internal mappings file for all instance types that are available at the time of release. You can add new mappings, or override existing mappings, by creating another custom mappings file. Only new or changed mappings need to be included in the custom mappings file.

The standard location for the custom mappings file is `etc/ec2.virtualizationmappings.properties` under the AWS plugin directory. An example file is provided in the `etc` directory as a basis for customization. You can provide a different location to Cloudera Director by setting the configuration property

`lp.ec2.virtualization.customMappingsPath` in one of the usual ways (in `application.properties` or on the command line). If the property is a relative path, it is based on the `etc` directory under the AWS plugin directory.

Here is an example of a custom mappings file that adds the new “d2” instance types introduced in AWS at the end of March 2015. These new instance types only support HVM virtualization. To keep the example short, many instance types are omitted; in an actual custom mappings file, each property value must provide the full list of instance types that support the property key and virtualization type.

```
hvm=m3.medium,\
m3.large,\
m3.xlarge,\
m3.2xlarge,\
...
d2.xlarge,\
d2.2xlarge,\
d2.4xlarge,\
d2.8xlarge
```

To learn more about virtualization types, see [Linux AMI Virtualization Types](#) in the AWS documentation.

Updated Ephemeral Device Mappings

Each AWS instance type provides zero or more instance store volumes, also known as ephemeral storage. These volumes are distinct from EBS-backed storage volumes; some instance types include no ephemeral storage. Cloudera Director specifies naming for each ephemeral volume, and keeps a list of the number of such volumes supported per instance type in an ephemeral device mappings file.

The AWS plugin included with Cloudera Director ships with an internal mappings file for all instance types that are available at the time of release. You can add new mappings, or override existing mappings, by creating another custom mappings file. Only new or changed mappings need to be included in the custom mappings file.

The standard location for the custom mappings file is `etc/ec2.ephemeraldevicemappings.properties` under the AWS plugin directory. An example file is provided in the `etc` directory as a basis for customization. You can provide a different location to Cloudera Director by setting the configuration property

`lp.ec2.ephemeral.customMappingsPath` in one of the usual ways (in `application.properties` or on the command line). If the property is a relative path, it is based on the `etc` directory under the AWS plugin directory.

Here is an example of a custom mappings file that describes the new “d2” instance types introduced at the end of March 2015. These new instance types each support a different number of instance store volumes.

```
d2.xlarge=3
d2.2xlarge=6
```

```
d2.4xlarge=12
d2.8xlarge=24
```

To learn more about ephemeral storage, including the counts for each instance type, see [Instance Stores Available on Instance Types](#) in the AWS documentation.

Using the New Mappings

Once the custom mappings files have been created, restart the Cloudera Director server so that they are detected and overlaid on the built-in mappings.

New instance types do not automatically appear in drop-down menus in the Cloudera Director web interface. However, the selected values for these menus may be edited by hand to specify a new instance type.

Post-creation Scripts

Post-creation scripts allow users to supply one or more scripts to be run after a Cloudera Manager cluster has been created. The supplied scripts are executed sequentially on a randomly selected cluster host. Scripts are not limited to Bash scripts, but can be written in any scripting language that can be interpreted on the system where it runs.

Configuring the Scripts

Post-creation scripts are available only through the command-line interface (CLI). There are two ways to supply post-creation scripts in the client configuration file:

- Use the `postCreateScripts` directive inside of the `cluster {}` configuration block. This block can take an array of scripts, and they can be entered directly into the configuration file, similar to the `bootstrapScript` that can be placed inside the `instance {}` configuration block.
- Use the `postCreateScriptsPaths` directive, also inside of the `cluster {}` configuration block. It can take an array of paths to arbitrary files on the local file system. This is similar to the `bootstrapScriptPath` directive. Cloudera Director will read the files from the file system and use these files' contents in their entirety as post-creation scripts.

Unlike `bootstrapScript` and `bootstrapScriptPath`, both ways can be used simultaneously. For example, the `postCreateScripts` directive can be used for setup (package installation, light system configuration), and the `postCreateScriptsPaths` directive can be used to refer to more complex scripts that may depend on the configuration that was performed in the `postCreateScripts` directive. Everything in the `postCreateScripts` block is executed sequentially first, and then everything in `postCreateScriptsPaths` is executed sequentially.

```
cluster {
  ....
  postCreateScripts: [#!/usr/bin/python]
  print 'Hello World Again!'

  #!/bin/bash
  echo 'Hello World!',

  postCreateScriptsPaths: ["/tmp/script1.py", "/tmp/script2.sh"]
}
```

Predefined Environment Variables

Post-creation scripts have access to several environment variables defined by Cloudera Director. Use these variables in your scripts to communicate with Cloudera Manager and configure it after Cloudera Director has completed its tasks.

Variable Name	Example	Description
DEPLOYMENT_HOST_PORT	192.168.1.100:7180	The host and port used to connect to the Cloudera Manager deployment that this cluster belongs to.
ENVIRONMENT_NAME	Cloudera Director Environment	The name of the environment that this cluster belongs to.
DEPLOYMENT_NAME	Cloudera Director Deployment	The name of the Cloudera Manager deployment that this cluster belongs to.
CLUSTER_NAME	Cloudera Director Cluster	The name of the cluster. The Cloudera Manager API will need this in order to specify which cluster on a Cloudera Manager server to operate on.
CM_USERNAME	admin	The username needed to connect to the Cloudera Manager deployment.
CM_PASSWORD	admin	The password needed to connect to the Cloudera Manager deployment.

Enabling Sentry Service Authorization

This topic describes how to enable the Sentry service with Cloudera Director.

Prerequisites

- Cloudera Director 1.1.x
- CDH 5.1.x (or higher) managed by Cloudera Manager 5.1.x (or higher).
- [Kerberos authentication](#) implemented on your cluster.

Setting Up the Sentry Service Using the Cloudera Director CLI

This method requires you to send configuration files that the Cloudera Director server can use to deploy clusters. See [Submitting a Cluster Configuration File](#) for more details. Make sure you add `SENTRY` to the array of `services` to be launched. This is specified in the configuration file as:

```
services: [HDFS, YARN, ZOOKEEPER, HIVE, OOZIE, HUE, IMPALA, SENTRY]
```

To specify a database, use the `databases` setting as follows:

```
cluster {
  ..
  databases {
    SENTRY: {
      type: mysql
      host: sentry.db.example.com
      port: 3306
      user: <database_username>
      password: <database_password>
      name: <database_name>
    }
  }
}
```

The Sentry service also requires the following custom configuration for the MapReduce, YARN, HDFS, Hive, and Impala Services.

- **MapReduce:** Set the **Minimum User ID for Job Submission** property to zero (the default is 1000) for every TaskTracker role group that is associated with Hive.

```
MAPREDUCE {
  TASKTRACKER {
    taskcontroller_min_user_id: 0
  }
}
```

- **YARN:** Ensure that the **Allowed System Users** property, for every NodeManager role group that is associated with Hive, includes the hive user.

```
YARN {
  NODEMANAGER {
    container_executor_allowed_system_users: hive, impala, hue
  }
}
```

- **HDFS:** Enable HDFS extended ACLs.

```
HDFS {
  dfs_permissions: true
  dfs_namenode_acls_enabled: true
}
```

With Cloudera Manager 5.3 and CDH 5.3, you can enable synchronization of HDFS and Sentry permissions for HDFS files that are part of Hive tables. For details on enabling this feature using Cloudera Manager, see [Synchronizing HDFS ACLs and Sentry Permissions](#).

- **Hive:** Make sure Sentry policy file authorization has been disabled for Hive.

```
HIVE {
  sentry_enabled: false
}
```

- **Impala:** Make sure Sentry policy file authorization has been disabled for Impala.

```
IMPALA {
  sentry_enabled: false
}
```

Set Permissions on the Hive Warehouse

Once setup is complete, configure the following permissions on the Hive warehouse. For Sentry authorization to work correctly, the Hive warehouse directory (`/user/hive/warehouse` or any path you specify as `hive.metastore.warehouse.dir` in your `hive-site.xml`) must be owned by the Hive user and group.

- Permissions on the warehouse directory must be set as follows:
 - **771** on the directory itself (for example, `/user/hive/warehouse`)
 - **771** on all subdirectories (for example, `/user/hive/warehouse/mysubdir`)
 - All files and subdirectories must be owned by `hive:hive`

For example:

```
$ sudo -u hdfs hdfs dfs -chmod -R 771 /user/hive/warehouse
$ sudo -u hdfs hdfs dfs -chown -R hive:hive /user/hive/warehouse
```

Setting up the Sentry Service Using the Cloudera Director API

You can use the Cloudera Director API to set up Sentry. Define the ClusterTemplate to include Sentry as a service, along with the configurations specified above, but in JSON format.

Customization and Advanced Configuration

Set permissions on the Hive warehouse as described [above](#).

Related Links

For detailed instructions on adding and configuring the Sentry service, see [Installing and Upgrading the Sentry Service](#) and [Configuring the Sentry Service](#).

Examples on using Grant/Revoke statements to enforce permissions using Sentry are available at [Hive SQL Syntax](#).

Upgrading Cloudera Director

This section contains notes and procedures for upgrading Cloudera Director.

Before Upgrading Cloudera Director from 1.1.x to 1.5.x

Follow these steps before upgrading to Cloudera Director 1.5.x.

1. Let running operations finish.

For example, if Cloudera Director is setting up a Cloudera Manager or CDH cluster (indicated by a progress bar in the UI), an upgrade will not complete successfully. An error in the log file instructs to use the old version of Cloudera Director until all running operations are completed, and then perform the upgrade.

2. Back up the Cloudera Director database that stores state information. By default, this is the embedded H2 database found at `/var/lib/cloudera-director-server/state.h2.db`.

If you are using a MySQL database to store Cloudera Director state, use MySQL backup procedures to back up the Cloudera Director database. The following example shows how to do this using the `mysqldump` utility:

```
mysqldump --all-databases --single-transaction --user=root --password > backup.sql
```

For more information on using `mysqldump`, see the [MySQL documentation](#).

3. Change your default encryption key.

After an upgrade to Cloudera Director 1.5.0 and higher, any new data that Cloudera Director persists in its database is encrypted with a default encryption key. For increased security, Cloudera recommends that you change your encryption key in the `application.properties` file before performing the upgrade. The file is located at `/etc/cloudera-director-server/application.properties`.

For more information about encryption and Cloudera Director data, see [Cloudera Director Database Encryption](#) on page 54.

Changes to the `application.properties` File in Cloudera Director 1.5.x

If you modified the `application.properties` file in Cloudera Director 1.1.x, the result of upgrading to Cloudera Director 1.5.x depends on which version of Linux you are using:

- **RHEL and CentOS** - New properties introduced in Cloudera Director 1.5 are added to `application.properties.rpmnew`. The original `application.properties` file functions as before and is not overwritten with the new Cloudera Director 1.5.x version properties. You do not need to copy the new properties from `application.properties.rpmnew` to the old `application.properties` file.
- **Debian and Ubuntu** - The modified Cloudera Director 1.1.x `application.properties` file is backed up to a file named `application.properties.dpkg-old`. The original `application.properties` file is then overwritten by the new `application.properties` file containing new Cloudera Director 1.5.x properties. After upgrading, copy your changes from `application.properties.dpkg-old` to the new `application.properties` file.

Upgrading Cloudera Director 1.1.x to Cloudera Director 1.5.x

The following sections describe steps for upgrading Cloudera Director 1.1.x to Cloudera Director 1.5.x on supported Linux operating systems.

RHEL and CentOS

1. Stop the Cloudera Director server service by issuing the following command:

```
sudo service cloudera-director-server stop
```

2. Open `/etc/yum.repos.d/cloudera-director.repo`. If the `baseurl` value in this file points to a specific minor or maintenance release version, like `/1.1/` or `/1.1.3/`, update the URL to point to the new 1.5.x version, such as `1.5.1`. If the URL points to `/1/`, you do not need to update this file.
3. Issue the following commands:

```
sudo yum clean all
sudo yum update cloudera-director-server cloudera-director-client
sudo service cloudera-director-server start
```

SLES

1. Stop the Cloudera Director server service by issuing the following command:

```
sudo service cloudera-director-server stop
```

2. Open `/etc/zypp/repos.d/cloudera-director.repo`. If the `baseurl` value in this file points to a specific minor or maintenance release version, like `/1.1/` or `/1.1.3/`, update the URL to point to the new 1.5.x version, such as `1.5.1`. If the URL points to `/1/`, you do not need to update this file.
3. Issue the following commands:

```
sudo zypper clean --all
sudo zypper update cloudera-director-server cloudera-director-client
sudo service cloudera-director-server start
```

Ubuntu

1. Stop the Cloudera Director server service by issuing the following command:

```
sudo service cloudera-director-server stop
```

2. Open `/etc/apt/sources.list.d/cloudera-director.list`. If the `baseurl` value in this file points to a specific minor or maintenance release version, like `/wheezy-director1.1/` or `/trusty-director1.1.3/`, update the URL to point to the new version, 1.5.x. If the URL points to `/1/`, you do not need to update this file.
3. Issue the following commands:

```
sudo apt-get clean
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get install cloudera-director-server cloudera-director-client
```

4. If your original Cloudera Director 1.1.x `application.properties` file has not been modified, proceed to the next step. If your `application.properties` file was modified in Cloudera Director 1.1.x, the original properties file will be overwritten by the new properties file containing new Cloudera Director 1.5.x properties, as described in [Upgrading Cloudera Director 1.1.x to Cloudera Director 1.5.x](#) on page 81. After completing steps 1–3, copy your changes from `application.properties.dpkg-old` to the new `application.properties` file before restarting the server.

5. Restart the Cloudera Director server:

```
sudo service cloudera-director-server start
```

Using IAM Policies with Cloudera Director 1.5 and Higher

In AWS, if you are using an IAM policy to control access to resources in the VPC, Cloudera Director 1.5 and higher requires permission for the method `DescribeDBSecurityGroups`. To give Cloudera Director permission for this method, add these values to your policy:

```
{
    "Action": [ "rds:DescribeDBSecurityGroups" ],
    "Effect": "Allow",
    "Resource": [ "*" ]
}
```

This permission is required because Cloudera Director 1.5 and higher includes early validation of RDS credentials when creating or updating an environment, whether or not RDS database servers are used.

For a sample IAM policy that includes this permission, see [Example IAM Policy](#) on page 49. For more information on AWS IAM, see the [IAM User Guide](#) in the AWS documentation.

Troubleshooting Cloudera Director

This topic contains information on issues, causes, and solutions for problems you might face when setting up, configuring, or using Cloudera Director.

Error Occurs if Tags Contain Unquoted Special Characters

If you use the configuration file with the `bootstrap` command to start Cloudera Director client, or use the `bootstrap-remote` command to set up a cluster with Cloudera Director server, and you have added a configuration to the config file that includes special characters without quotes, you may receive an error. This applies to HOCON characters, and includes periods. If the added configuration is in the form `x.y`, for example, the following error message may be displayed: `"com.typesafe.config.ConfigException$WrongType: ... <x> has type OBJECT rather than STRING"`. This means that `x.y` must be in quotes, as in `"x.y"`. An example of a configuration that would require quoting is `"log.dirs"` in Kafka.

Viewing Cloudera Director Logs

To help you troubleshoot problems, you can view the Cloudera Director logs. Log files can be found in the following locations:

- Cloudera Director Client
 - One shared log file per user account:

```
$HOME/.cloudera-director/logs/application.log
```

- Cloudera Director Server
 - One file for all clusters:

```
/var/log/cloudera-director-server/application.log
```

Location of H2 Embedded Database

Cloudera Director uses an H2 embedded database to store environment and cluster data. The H2 embedded database file is located at:

```
/var/lib/cloudera-director-server/state.h2.db
```

DNS Issues

Symptom

Director fails to bootstrap a cluster with a DNS error.

Cause

The Amazon Virtual Private Cloud (VPC) is not set up for forward and reverse hostname resolution. Functional forward and reverse DNS resolution is a key requirement for many components of the Cloudera EDH platform, including Cloudera Director.

Solution

Configure the VPC for forward and reverse hostname resolution. You can verify if DNS is working as expected on a host by issuing the following one-line Python command:

```
python -c "import socket; print socket.getfqdn(); print socket.gethostbyname(socket.getfqdn())"
```

For more information on DNS and Amazon VPCs, see [DHCP Options Sets](#) in the Amazon VPC documentation.

If you are using Amazon-provided DNS, perform these steps to configure DHCP options:

1. Log in to the [AWS Management Console](#).
2. Select **VPC** from the **Services** navigation list box.
3. In the left pane, click **Your VPCs**. A list of currently configured **VPCs** appears.
4. Select the **VPC** you are using and note the **DHCP options set ID**.
5. In the left pane, click **DHCP Option Sets**. A list of currently configured DHCP Option Sets appears.
6. Select the option set used by the VPC.
7. Check for an entry similar to the following and make sure the domain-name is specified. For example:

```
domain-name = ec2.internal
domain-name-servers = AmazonProvidedDNS
```



Note: If you're using AmazonProvidedDNS in us-east-1, specify `ec2.internal`. If you're using AmazonProvidedDNS in another region, specify `region.compute.internal` (for example, `ap-northeast-1.compute.internal`).

8. If it is not configured correctly, create a new DHCP option set for the specified region and assign it to the VPC. For information on how to specify the correct domain name, see the [AWS Documentation](#).

Server doesn't start

Symptom

The Cloudera Director server doesn't start or quickly exits with an Out of Memory exception.

Cause

The Cloudera Director server is running on a machine with insufficient memory.

Solution

Run Cloudera Director on an instance that has at least 1GB of free memory. See [Requirements and Supported Versions](#) on page 16 for more details on Cloudera Director hardware requirements.

Problem When Removing Hosts from a Cluster

Symptom

A **Modify Cluster** operation fails to complete.

Troubleshooting Cloudera Director

Cause

You are trying to shrink the cluster below the HDFS replication factor. See [Removing or Repairing Hosts in a Cluster](#) on page 42 for more information about replication factors.

Solution

Do not attempt to shrink a cluster below the HDFS replication factor. Doing so can result in a loss of data.

Problems Connecting to Cloudera Director Server

Symptom

You are unable to connect to the Cloudera Director server.

Cause

Configuration of security group and iptables settings. For more information about configuring security groups and turning off iptables, see [Setting Up a VPC](#) on page 20. Some operating systems have IP tables turned on by default, and they must be turned off.

Solution

Check security group and iptables settings and reconfigure if necessary.

Frequently Asked Questions

This page answers frequently asked questions about Cloudera Director.

General Questions

How can I reduce the time required for cluster deployment?

You can reduce cluster deployment time by using an Amazon Machine Image (AMI). For information on creating an AMI, see [Creating a Cloudera Manager and CDH AMI](#) on page 47.

How can I find a list of available AMIs?

Perform the following steps to generate a list of RHEL 64-bit images:

1. Install the AWS CLI.

```
$ sudo pip install awscli
```

2. Configure the AWS CLI.

```
$ aws configure
```

Follow the prompts. Choose any output format. The following example command defines *table* as the format.

3. Run the following query:

```
aws ec2 describe-images \  
--output table \  
--query 'Images[*].[VirtualizationType,Name,ImageId]' \  
--owners 309956199498 \  
--filters \  
  Name=root-device-type,Values=ebs \  
  Name=image-type,Values=machine \  
  Name=is-public,Values=true \  
  Name=hypervisor,Values=xen \  
  Name=architecture,Values=x86_64
```

AWS returns a table of available images in the region you configured.

Cloudera Director Glossary

availability zone

A distinct location in the region that is insulated from failures in other availability zones. For a list of regions and availability zones, see [Regions and Availability Zones](#) in the AWS documentation.

Cloudera Director

An application for deploying and managing CDH clusters using configuration template files.

Cloudera Manager

An end-to-end management application for CDH clusters. Cloudera Manager enables administrators to easily and effectively provision, monitor, and manage Hadoop clusters and CDH installations.

cluster

A set of computers that contains an HDFS file system and other CDH components.

cluster launcher

An instance that launches a cluster using Cloudera Director and the configuration file.

configuration file

A template file used by Cloudera Director that you modify to launch a CDH cluster.

deployment

See cluster. Additionally, deployment refers to the process of launching a cluster.

environment

The region, account credentials, and other information used to deploy clusters in a cloud infrastructure provider.

ephemeral cluster

A short lived cluster that launches, processes a set of data, and terminates. Ephemeral clusters are ideal for periodic jobs.

instance

One virtual server running in a cloud environment, such as AWS.

instance group

A specification that includes general instance settings (such as the instance type and role settings), which you can use to launch instances without specifying settings for each individual instance.

instance type

A specification that defines the memory, CPU, storage capacity, and hourly cost for an instance.

keys

The combination of your AWS access key ID and secret access key used to sign AWS requests.

long-lived cluster

A cluster that remains running and available.

provider

A company that offers a cloud infrastructure which includes computing, storage, and platform services. Providers include AWS, Rackspace, and HP Public Cloud.

region

A distinct geographical AWS data center location. Each region contains at least two availability zones. For a list of regions and availability zones, see <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html>.

tags

Metadata (name/value pairs) that you can define and assign to instances. Tags make it easier to find instances using environment management tools. For example, AWS provides the AWS Management Console.

template

A template file that contains settings that you use to launch clusters.