

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 2.1.x

Date: October 13, 2016

Table of Contents

Introduction.....	8
Cloudera Director Features.....	8
Cloudera Director Client and Server.....	9
<i>Cloudera Director Client.....</i>	<i>9</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 2.....</i>	<i>11</i>
<i>New Features and Changes in Cloudera Director 1.....</i>	<i>11</i>
Known Issues and Workarounds in Cloudera Director.....	12
<i>Cannot update environment credentials of environments deployed on Microsoft Azure.....</i>	<i>12</i>
<i>External databases are not configured for Hue and Oozie.....</i>	<i>13</i>
<i>Cloudera Director does not install the JDBC driver for an existing MySQL database.....</i>	<i>13</i>
<i>Cloudera Director cannot deploy Cloudera Navigator Key Trustee Server.....</i>	<i>13</i>
<i>Creation of a cluster where instance groups have no roles is not possible using the web UI.....</i>	<i>14</i>
<i>Modification of a cluster where instance groups have no roles is not possible using the web UI.....</i>	<i>14</i>
<i>Resize script cannot resize XFS partitions.....</i>	<i>14</i>
<i>Cloudera Director does not set up external databases for Sqoop2.....</i>	<i>14</i>
<i>Metrics not displayed for clusters deployed in Cloudera Manager 5.4 and earlier clusters.....</i>	<i>14</i>
<i>Changes to Cloudera Manager username and password must also be made in Cloudera Director.....</i>	<i>14</i>
<i>Cloudera Director does not sync with cluster changes made in Cloudera Manager.....</i>	<i>14</i>
<i>Cloudera Director may use AWS credentials from instance of Cloudera Director server.....</i>	<i>14</i>
<i>Root partition resize fails on CentOS 6.5 (HVM).....</i>	<i>15</i>
<i>Terminating clusters that are bootstrapping must be terminated twice for the instances to be terminated.....</i>	<i>15</i>
<i>When using RDS and MySQL, Hive Metastore canary may fail in Cloudera Manager.....</i>	<i>15</i>
Issues Fixed in Cloudera Director.....	15
<i>Issues Fixed in Cloudera Director 2.1.1.....</i>	<i>15</i>
<i>Issues Fixed in Cloudera Director 2.1.0.....</i>	<i>16</i>
<i>Issues Fixed in Cloudera Director 2.0.0.....</i>	<i>17</i>
<i>Issues Fixed in Cloudera Director 1.5.2.....</i>	<i>17</i>
<i>Issues Fixed in Cloudera Director 1.5.1.....</i>	<i>18</i>
<i>Issues Fixed in Cloudera Director 1.5.0.....</i>	<i>18</i>
<i>Issues Fixed in Cloudera Director 1.1.3.....</i>	<i>18</i>
<i>Issues Fixed in Cloudera Director 1.1.2.....</i>	<i>18</i>
<i>Issues Fixed in Cloudera Director 1.1.1.....</i>	<i>19</i>

Requirements and Supported Versions.....20

Cloud Providers.....20

Cloudera Director Service Provider Interface (SPI).....20

Supported Software and Distributions.....20

Resource Requirements.....21

Supported Cloudera Manager and CDH Versions.....21

Networking and Security Requirements.....22

Ports Used by Cloudera Director.....22

Supported Browsers.....23

Getting Started with Cloudera Director.....24

Getting Started on Amazon Web Services (AWS).....24

Setting up the AWS Environment.....24

Launching an EC2 Instance for Cloudera Director.....25

Installing Cloudera Director Server and Client on the EC2 Instance.....27

Configuring a SOCKS Proxy for Amazon EC2.....30

Deploying Cloudera Manager and CDH on AWS.....32

Cleaning Up Your AWS Deployment.....37

Getting Started on Google Cloud Platform.....37

Creating a Google Cloud Platform Project.....37

Configuring Tools for Your Google Cloud Platform Account.....38

Creating a Google Compute Engine VM Instance.....39

Installing Cloudera Director Server and Client on Google Compute Engine.....40

Configuring a SOCKS Proxy for Google Compute Engine.....43

Deploying Cloudera Manager and CDH on Google Compute Engine.....43

Cleaning Up Your Google Cloud Platform Deployment.....48

Getting Started on Microsoft Azure.....48

Obtaining Credentials for Cloudera Director.....48

Setting up Azure Resources.....48

Setting Up Dynamic DNS on Azure.....53

Setting Up MySQL or PostgreSQL.....60

Setting Up a Virtual Machine for Cloudera Director Server.....60

Installing Cloudera Director Server and Client on Azure.....61

Configuring a SOCKS Proxy for Microsoft Azure.....62

Allowing Access to VM Images.....63

Creating a Cluster.....64

Adding New VM Images.....71

Important Notes.....71

Usage-Based Billing.....73

Prerequisites.....	73
How Usage-Based Billing Works.....	73
Deploying Cloudera Manager and CDH with Usage-Based Billing.....	74
<i>Enabling Usage-Based Billing with the Cloudera Director Server web UI.....</i>	<i>74</i>
<i>Enabling Usage-Based Billing with bootstrap-remote.....</i>	<i>74</i>
Managing Billing IDs with an Existing Deployment.....	75
Troubleshooting Network Connectivity for Usage-Based Billing.....	76

Customization and Advanced Configuration.....77

The Cloudera Director Configuration File.....	77
<i>Location of Sample Configuration Files.....</i>	<i>77</i>
<i>Customizing the Configuration File.....</i>	<i>77</i>
<i>Valid Role Types for Use in Configuration Files.....</i>	<i>77</i>
Using Spot Instances.....	77
<i>Planning for Spot Instances.....</i>	<i>77</i>
<i>Specifying Spot Instances.....</i>	<i>78</i>
<i>Best Practices for Using Spot Instances.....</i>	<i>78</i>
Creating a Cloudera Manager and CDH AMI.....	78
Choosing an AMI.....	78
<i>Finding Available AMIs.....</i>	<i>79</i>
Running Cloudera Director and Cloudera Manager in Different Regions or Clouds.....	79
Deploying a Java 8 Cluster.....	80
<i>javaInstallationStrategy configuration.....</i>	<i>80</i>
<i>Bootstrap script.....</i>	<i>80</i>
Creating AWS Identity and Access Management (IAM) Policies.....	81
Using MySQL for Cloudera Director Server.....	83
<i>Installing the MySQL Server.....</i>	<i>83</i>
<i>Configuring and Starting the MySQL Server.....</i>	<i>83</i>
<i>Installing the MySQL JDBC Driver.....</i>	<i>85</i>
<i>Creating a Database for Cloudera Director Server.....</i>	<i>86</i>
<i>Configuring Cloudera Director Server to use the MySQL Database.....</i>	<i>86</i>
Using MariaDB for Cloudera Director Server.....	87
<i>Installing the MariaDB Server.....</i>	<i>87</i>
<i>Configuring and Starting the MariaDB Server.....</i>	<i>87</i>
<i>Installing the MariaDB JDBC Driver.....</i>	<i>89</i>
<i>Creating a Database for Cloudera Director Server.....</i>	<i>90</i>
<i>Configuring Cloudera Director Server to use the MariaDB Database.....</i>	<i>90</i>
Cloudera Director Database Encryption.....	91
<i>Cipher Configuration.....</i>	<i>91</i>
<i>Starting with Encryption.....</i>	<i>92</i>
<i>Changing Encryption.....</i>	<i>93</i>
Using an External Database for Cloudera Manager and CDH.....	94

<i>Defining External Database Servers</i>	94
<i>Defining External Databases</i>	100
Setting Cloudera Director Properties.....	103
Setting Cloudera Manager Configurations.....	111
<i>Cluster Configuration Using Cloudera Manager</i>	112
<i>Setting up a Cloudera Manager License</i>	112
<i>Deployment Template Configuration</i>	113
<i>Cluster Template Service-wide Configuration</i>	114
<i>Cluster Template Roletype Configurations</i>	114
Configuring Cloudera Director for a New AWS Instance Type.....	115
<i>Updated Virtualization Mappings</i>	115
<i>Updated Ephemeral Device Mappings</i>	116
<i>Using the New Mappings</i>	116
Modifying or Updating Clusters Using Cloudera Manager.....	116
Post-Creation Scripts.....	118
<i>Configuring the Scripts</i>	118
<i>Predefined Environment Variables</i>	118
Creating Kerberized Clusters With Cloudera Director.....	119
<i>Creating a Kerberized Cluster with the Cloudera Director Configuration File</i>	119
Creating Highly Available Clusters With Cloudera Director.....	120
<i>Limitations and Restrictions</i>	121
<i>Editing the Configuration File to Launch a Highly Available Cluster</i>	121
<i>Using Role Migration to Repair HDFS Master Role Instances</i>	123
Enabling Sentry Service Authorization.....	124
<i>Prerequisites</i>	124
<i>Setting Up the Sentry Service Using the Cloudera Director CLI</i>	124
<i>Setting up the Sentry Service Using the Cloudera Director API</i>	126
<i>Related Links</i>	126

Managing Cloudera Manager Instances with Cloudera Director Server.....127

Submitting a Cluster Configuration File.....	127
Deploying Clusters in an Existing Environment.....	127
Cloudera Manager Health Information.....	128
Opening Cloudera Manager.....	129
Creating and Modifying Clusters with the Cloudera Director web UI.....	129
<i>Configuring Instance Groups During Cluster Creation</i>	129
<i>Modifying the Number of Instances in an Existing Cluster</i>	130
<i>Repairing Worker and Gateway Instances in a Cluster</i>	132
Terminating a Cluster.....	132
<i>Terminating a Cluster with the web UI</i>	132
<i>Terminating a Cluster with the CLI</i>	133
Starting and Stopping the Cloudera Director Server.....	133

User Management.....	133
<i>Managing Users with the Cloudera Director Web web UI.....</i>	<i>133</i>
<i>Managing Users with the Cloudera Director API.....</i>	<i>134</i>

Cloudera Director Client.....136

Installing Cloudera Director Client.....	136
Provisioning a Cluster on AWS.....	137
Running Cloudera Director Client.....	138
Using the Command Line Interface.....	139
<i>Local commands.....</i>	<i>139</i>
<i>Remote commands.....</i>	<i>140</i>
Connecting to Cloudera Manager with Cloudera Director Client.....	140
Modifying a Cluster with the Configuration File.....	142
<i>Growing or Shrinking a Cluster with the Configuration File.....</i>	<i>142</i>

Upgrading Cloudera Director.....143

Before Upgrading Cloudera Director.....	143
Upgrading Cloudera Director.....	144

Troubleshooting Cloudera Director.....147

Cloudera Director Cannot Manage a Cluster That Was Kerberized Through Cloudera Manager.....	147
New Cluster Fails to Start Because of Missing Roles.....	148
Cloudera Director Server Will Not Start with Unsupported Java Version.....	148
Error Occurs if Tags Contain Unquoted Special Characters.....	148
DNS Issues.....	149
Server Does Not Start.....	150
Problem When Removing Hosts from a Cluster.....	150
Problems Connecting to Cloudera Director Server.....	150

Frequently Asked Questions.....151

General Questions.....	151
------------------------	-----

Cloudera Director Glossary.....152

Introduction

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

Cloudera Director provides 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.

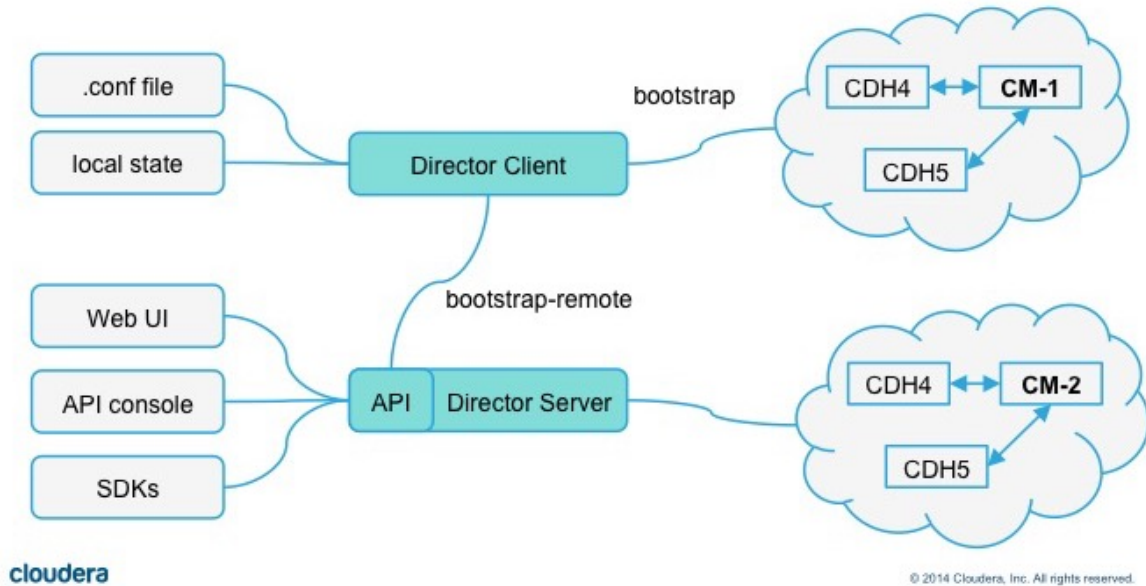
Benefit	Features
Simplified cluster lifecycle management	Simple user interface: <ul style="list-style-type: none"> • Self-Service spin up and tear down • Scaling of clusters for spiky workloads • Simple cloning of clusters • Cloud blueprints for repeatable deployments
Elimination of lock-in	Flexible, 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

Benefit	Features
	<ul style="list-style-type: none"> 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 web 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 10. For more information about the `.conf` file, see [The Cloudera Director Configuration File](#) on page 77.

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 web 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 Web User Interface (web UI) Only

The web 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 web UI for initial setups, and interact with Cloudera Director server through the web 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 web UI provides, and not for customized setups. Although many advanced features are available using only the web 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 web 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 send 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 web UI mode and entering the settings preserved in this `.conf` file.

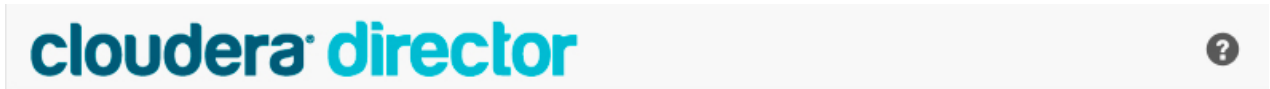
For maximum flexibility and power—for example, if you want to experiment with custom configurations and custom role assignments—using `bootstrap-remote` with the `.conf` file is a good choice.

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 web 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 new features and known issues and limitations for Cloudera Director.

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

New Features and Changes in Cloudera Director

New Features and Changes in Cloudera Director 2

The following sections describe what's new and changed in each Cloudera Director 2 release.

What's New in Cloudera Director 2.1.1

- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 2.1.1](#) for details.
- Cloudera Director now includes support for the following Microsoft Azure instance types:
 - Standard_DS12_v2
 - Standard_DS13_v2
 - Standard_DS14_v2
- On Microsoft Azure, Cloudera Director now supports the RHEL 6.7 image published by Red Hat in partnership with Microsoft.

What's New in Cloudera Director 2.1.0

- Usage-based billing, where the cost of running a cluster is based on cluster usage, is supported. See [Usage-Based Billing](#) on page 73.
- Running CM and CDH clusters on Microsoft Azure is supported. See [Getting Started on Microsoft Azure](#) on page 48
- Deploying Cloudera Manager and CDH on a different cloud provider or region than Cloudera Director with a simple network setup is supported. See [Running Cloudera Director and Cloudera Manager in Different Regions or Clouds](#) on page 79.
- Cloudera 5.7 is supported out of the box.
- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 2.1.0](#) for details.

What's New in Cloudera Director 2.0.0

- AWS Spot Instances and Google Cloud Platform Preemptible Instances are supported.
- Setup of clusters that are highly available and authenticated through Kerberos is automated.
- You can automate submission of jobs to clusters with dynamic creation and termination of clusters.
- You can run custom scripts after cluster setup and before cluster termination.
- The user interface is enhanced, with deeper insights into cluster health.
- Reliability of cluster modifications is increased, including rollback in some failure scenarios.
- RHEL 7.1 is supported.
- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 2.0.0](#) for details.

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.

Cloudera Director Release Notes

- 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 18 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 through 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 web UI.
- You 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 through the web UI.
- A number of issues have been fixed. See [Issues Fixed in Cloudera Director 1.5.0](#) on page 18 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 18 for details.
- The Cloudera Director 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

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

Cannot update environment credentials of environments deployed on Microsoft Azure

With Cloudera Director on Microsoft Azure, the **Update Environment Credentials web UI** displays only some properties, and does not display all the properties required for the update.

Workaround: Use the API console to update the credentials by performing the following steps:

1. Go to the **Environments** section of the API console.
2. Open the **Get /api/v5/environments** section, and click **Try it out!** to list the environments.

3. Copy the environment name for the environment in which you want to update the credentials.
4. Open the **Get** `/api/v5/environments/{name}` section, paste the environment name into the name parameter box, and click **Try it out!** to display the environment.
5. Copy the `config` block of the JSON (should have keys like `tenantId`, etc.). You only want the curly braces and content, not the `config:` key.
6. Open the **Put** `/api/v5/environments/{name}/provider/credentials` section, paste the JSON block, remove the line containing the region (see **Note** below for an explanation), and replace the `REDACTED` value of `clientSecret` with the new value.
7. Click **Try it out!** to update the credentials.

On success, you should get a 202 response code. For other response codes, check the error message and look in the Cloudera Director server log if necessary.



Note: The region key is not a credentials configuration key. You can see all the valid credentials configuration keys by opening the provider-metadata section of the API console, opening the **Get** `/api/v5/metadata/providers/{providerId}` section, entering `azure` in the **providerId** parameter box, clicking **Try it out!** to list the metadata, and finding the `credentialsProperties` JSON block. The `configKey` field of the entries in there are the valid credentials property configuration keys.

External databases are not configured for Hue and Oozie

External databases are not configured for Hue and Oozie in clusters created through the Cloudera Director web UI.

Workaround: External databases can be specified for Hue and Oozie when creating a cluster using the CLI or API.

Cloudera Director does not install the JDBC driver for an existing MySQL database

Cloudera Director automatically installs JDBC drivers on an instance for Cloudera Manager and the CDH clusters it provisions. However, when you use an existing MySQL database with Cloudera Manager, Cloudera Director does not install the JDBC driver, which can result in database connection failures.

Workaround: Select an instance image that already has the JDBC driver installed or use the instance bootstrap script to install the latest available JDBC driver for MySQL.

To install the MySQL JDBC driver, download the driver packages for your platform:

OS	Packages
RHEL 5 or 6	<pre>mysql-connector-java mysql-devel</pre>
Ubuntu or Debian	<pre>libmysql-java libmysqlclient-dev</pre>

After you download the JDBC driver packages, you can use a package manager to install the packages.

Package Manager	Command
yum	<code>sudo yum install -d 1 --assumeyes package_name</code>
Apt	<code>sudo aptitude --show-deps --log-level=info --without-recommends --show-versions --verbose --assume-yes install package_name</code>

Cloudera Director cannot deploy Cloudera Navigator Key Trustee Server

Cloudera Navigator Key Trustee Server cannot be one of the services deployed by Cloudera Director.

Workaround: Contact Cloudera Support if you need to add Cloudera Navigator Key Trustee Server.

Creation of a cluster where instance groups have no roles is not possible using the web UI

Cloudera Director's web UI does not allow creation of clusters with instance groups that should not have CDH roles deployed on them.

Workaround: Use the CLI or API to create clusters with instance groups that should not have CDH roles deployed on them. At least one instance group in the cluster must specify roles. Otherwise roles will be automatically allocated across all instances.

Modification of a cluster where instance groups have no roles is not possible using the web UI

Cloudera Director's web UI does not allow modification of clusters with instance groups that should not have CDH roles deployed on them, even if they were created using the API.

Workaround: Use the API to modify clusters with instance groups that should not have CDH roles deployed on them.

Resize script cannot resize XFS partitions

Cloudera Director is unable to resize XFS partitions, which makes creating an instance that uses the XFS filesystem fail during bootstrap.

Workaround: Use an image with an ext filesystem such as ext2, ext3, or ext4.

Cloudera Director does not set up external databases for Sqoop2

Cloudera Director cannot set up external databases for Sqoop2.

Workaround: Set up databases for this service as described in [Cloudera Manager and Managed Service Databases](#).

Metrics not displayed for clusters deployed in Cloudera Manager 5.4 and earlier clusters

Clusters deployed in Cloudera Manager version 5.4 and lower might not have metrics displayed in the web UI if these clusters share the same name as previously deleted clusters.

Workaround: Use Cloudera Manager 5.5 and higher.

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 Cloudera Director web UI to update the Cloudera Manager username and password.

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 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.

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 web 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.

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 higher that is exposed when used with the MySQL JDBC driver (used by Cloudera Director) version 5.1.19 or lower. 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.

Issues Fixed in Cloudera Director

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

Issues Fixed in Cloudera Director 2.1.1

Cloudera Director cannot connect to restarted VMs on Azure

Restarted VMs on Microsoft Azure are sometimes assigned a new IP address. This causes the cached IP address in Cloudera Director to become stale, so that Cloudera Director is unable to connect to the VMs. Affected version: Cloudera Director 2.1.0.

Public IP attached to a VM on Azure is deleted when the VM is deleted

Any public IP attached to a VM is deleted when the VM is deleted, even if that public IP was not created by the plugin. Affected version: Cloudera Director 2.1.0.

Cloudera Director web UI handles errors incorrectly with failed instance template validation on Azure

When the Microsoft Azure subscription permissions are not properly set up, an unexpected error can occur, causing instance template validation to exit. This error is not properly displayed in the Cloudera Director web UI. Affected version: Cloudera Director 2.1.0.

Resource name cannot contain special characters

A deployment may fail if the compute resource group used for Azure deployment contains special characters such as an underscore (`_`). Resource group names are sometimes used in the construction of resource names, causing deployments to fail if the resource group names contain special characters, because the naming restrictions are different for resource group names and resource names. Affected version: Cloudera Director 2.1.0.

Bootstrapping of clusters may fail if configured to not associate public IP addresses with EC2 instances

When using AWS, if the user deselects the **Associate public IP addresses** checkbox, instructing Cloudera Director to not assign public IP addresses to the EC2 instances it creates, Cloudera Director incorrectly interprets the missing public

IP address of each instance as `localhost` (the Cloudera Director instance itself). Under certain conditions, this can lead to a variety of errors, including bootstrap failures and corruption of the Cloudera Director instance. Affected version: Cloudera Director 2.1.0.

Database server password fails if it contains special characters

Cloudera Director server does not handle special characters properly in database server admin/root passwords.

Update Cloudera Manager Credentials fails in certain scenarios

Cloudera Director erroneously rejects the credentials update as an unsupported modification if sensitive fields are configured on the deployment. The sensitive fields include `license`, `billingId`, and `krbAdminPassword`.

Cloudera Director server fails to start after upgrade under some circumstances

During an upgrade, Cloudera Director expects the Cloudera Manager instances it has deployed to match the instance template that was used while bootstrapping those instances. If the instance was modified out of band of Cloudera Director, then the server fails to start. An example of a mismatch is if the instance type of the Cloudera Manager instance was modified from within the cloud provider console.

Cluster bootstrap fails with high task parallelism

For high values of `lp.bootstrap.parallelBatchSize`, Cloudera Director fails to bootstrap clusters and throws an exception indicating that it failed to write intermediate state to the database. The default value of `lp.bootstrap.parallelBatchSize` is 20. `lp.bootstrap.parallelBatchSize` controls how many operations Cloudera Director should do in parallel while configuring a cluster.

Modifying a cluster can leave some roles marked as stale in Cloudera Manager

When growing or shrinking a cluster, you are presented with the option of restarting the cluster. The restart operation should only restart roles that are marked stale by Cloudera Manager, that is, only roles that need to be restarted. This optimization serves to minimize cluster downtime. However, with Cloudera Director 2.1.x, some stale roles might not be restarted, even though the **Restart Cluster** option is selected.

Default memory autoconfiguration for monitoring services may be suboptimal

Depending on the size of your cluster and your 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.

Issues Fixed in Cloudera Director 2.1.0

Validation error after initial setup with high availability

When you set up HDFS high availability using Cloudera Director, the secondary NameNode is not configured, because it is not required for high availability. Because of a Cloudera Manager bug, the absence of a secondary NameNode causes an erroneous validation error to appear in Cloudera Manager in **HDFS > Configuration > HDFS Checkpoint Directories**.

Repository or parcel URLs with internal domain names fail validation

Repository or parcel URLs fail validation in Cloudera Director when they are specified with internal domain names.

Database-related error when running Cloudera Director CLI after upgrade

When run after upgrade, the Cloudera Director CLI performs steps to upgrade its local database from the previous version. It can report an error:

```
Referential integrity management for DEFAULT not implemented.
```


Cloudera Director Does Not Recognize Cloudera Manager Password Changes

Cloudera Director does not recognize changes in the `admin` password in Cloudera Manager unless the username associated with the new password is also changed.

Incorrect yum repo definitions for Google Compute Engine RHEL images

The default RHEL 6 image defined in `director-google-plugin` version 1.0.1 and lower has an incorrect yum repo definition. This causes yum commands to fail after yum caches are cleared. See the [Google Compute Engine issue tracker](#) for issue details.

Long version string required for Kafka

Kafka requires a nonintuitive version string to be specified in the configuration file or web UI.

Issues Fixed in Cloudera Director 2.0.0

Cloning and growing a Kerberos-enabled cluster fails

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

Kafka with Cloudera Manager 5.4 and lower causes failure

Kafka installed with Cloudera Manager 5.4 and lower causes the Cloudera Manager installation wizard, and therefore the bootstrap process, to fail, unless you override the configuration setting `broker_max_heap_size`.

Cloudera Director does not set up external databases for Oozie and Hue

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

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 run 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.

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 does not start automatically after first run

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

Impala daemons attempt to connect over IPv6

Impala daemons attempt to connect over IPv6.

DNS queries occasionally time out with AWS VPN

DNS queries occasionally time out with AWS VPN.

Issues Fixed in Cloudera Director 1.1.3

Ensure accurate time on startup

Instance normalization has been improved to ensure that time is synchronized by Network Time Protocol (NTP) before 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 preinstalled Cloudera Manager packages

Cloudera Director no longer upgrades preinstalled 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 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 web 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), Google Cloud Platform, and Microsoft Azure.

Each Cloudera Director release embeds the current plug-in for supported cloud providers, but a newer plug-in 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 plug-in](#)
- [Google Cloud Platform cloud provider plug-in](#)
- [Microsoft Azure cloud provider plug-in](#)

Cloudera Director Service Provider Interface (SPI)

The Cloudera Director SPI defines an open source Java interface that plug-ins implement to add support for additional cloud providers to Cloudera Director. For more information, see the README.md file in 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 6.5, 6.7, 7.1, and 7.2 Ubuntu 14.04	For AWS and Google Cloud Platform: RHEL and CentOS 6.5, 6.7, 7.1, and 7.2 For Microsoft Azure: RHEL and CentOS 6.7 and 7.2 RHEL 7.2 is supported only for Cloudera Manager and CDH 5.7 and higher, not for lower versions of Cloudera Manager and CDH. To use Amazon EC2 D2 instances, you must run a minimum version of RHEL 6.7 or CentOS 6.7. Earlier versions of RHEL and CentOS do not support these instance types.
Oracle Java SE Development Kit (JDK)	Oracle JDK version 7 or 8 For download and installation information, see Java SE Downloads .	Oracle JDK version 7 or 8
Default Database	Embedded H2 database	Embedded PostgreSQL Database
Supported Databases	MySQL 5.5, 5.6, 5.7	MySQL 5.5, 5.6, 5.7

	Cloudera Director	Cloudera Manager and CDH
	MariaDB 5.5	MariaDB 5.5



Note: For the latest information on operating system versions supported on Microsoft Azure, refer to the [Cloudera Reference Architecture for Microsoft Azure Deployments](#).



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 83. 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 or c4.large	Cloudera Manager: m4.xlarge or m4.4xlarge
Recommended Google Cloud Platform instance	n1-standard-2	n1-highmem-4 or n1-highmem-8
Recommended Microsoft Azure instance	Standard_D3 or larger	The following Azure instance types are supported: <ul style="list-style-type: none"> Standard_DS12_v2 Standard_DS13_v2 Standard_DS14_v2 Standard_DS13 Standard_DS14



Note: For the latest information on instance types supported on Microsoft Azure, refer to the [Cloudera Reference Architecture for Microsoft Azure Deployments](#).



Note: The recommended instance for Cloudera Manager depends on the workload. Some instance types may not be available in every region. Cloudera Director does not dynamically validate instance type by region. Contact your Cloudera account representative for more information.

Supported Cloudera Manager and CDH Versions

Cloudera Director 2.1 can install any version of Cloudera Manager 5 with any CDH 5 parcels. Use of CDH packages is not supported.

Requirements and Supported Versions

If you are using Cloudera Director 2.1 to deploy Cloudera Manager and CDH, the latest released version of Cloudera Manager 5.7 and CDH 5.7 is installed by default. To use Cloudera Manager 5.8 and CDH 5.8, follow the instructions for installing non-default versions of Cloudera Manager and CDH in the Getting Started section for your cloud provider:

- For AWS, see [Deploying Cloudera Manager and CDH on AWS](#) on page 32.
- For Google Cloud Platform, see [Deploying Cloudera Manager and CDH on Google Compute Engine](#) on page 43.
- For Microsoft Azure, see [Deploying Cloudera Manager and CDH on Microsoft Azure](#).

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.



Note: The **All traffic** rule above requires the security group ID. If you create a security group from scratch, create the security group with the SSH rule and then go back and edit the security group to allow all traffic within the security group.

To connect to the AWS network, Cloudera recommends that you open only these ports and set up a SOCKS proxy. Unless your network has direct connection to AWS, you must set this up to access the Cloudera Director instance. This is done in a later step.

Ports Used by Cloudera Director

Cloudera Director uses the ports listed in the table below.

Component	Service	Port	Access Requirement	Configuration	Comment
Cloudera Director server	HTTP/HTTPS port for Cloudera Director web UI and API	7189	Must be accessible from outside the cluster	<code>server.port</code> in <code>application.properties</code>	Web web UI and API
Cloudera Director internal shell	CRaSH shell port	2000	localhost only	<code>shell.ssh.port</code> in <code>application.properties</code>	Used with the ssh client

Cloudera Director also uses the following ports in a typical deployment:

- 80 and 443: to connect to external services for validation and tracking.
- 7180: to talk to the Cloudera Manager API
- 22: to connect to new instances over SSH
- 123: to configure NTP within the cluster.

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) and Google Cloud Platform.

Getting Started on Amazon Web Services (AWS)

To use Cloudera Director on AWS, you 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 describes the steps for each of these tasks.



Important:

Cloudera Director supports Spot instances. Spot instances are virtual machines that have a lower cost but are subject to reclamation at any time by AWS. Because of the possibility of interruption, Cloudera recommends that you use Spot instances only for worker roles in a cluster, not for master or gateway roles. Cloudera Director only supports Spot instances for CentOS.

For more information about using Spot instances with Cloudera Director, see [Using Spot Instances](#) on page 77.

Setting up the AWS Environment

You must set up a VPC and create an SSH key pair in the AWS environment before deploying Cloudera Director.

Setting Up a VPC

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

To set up a new VPC, follow the steps below. Skip these steps if you are using an existing VPC.

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. For the easiest way to get started, select **VPC with a Single Public Subnet**.
5. Complete the VPC wizard and then click **Create VPC**.

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. You can configure Cloudera Director to use proxy servers or a local mirror of all the relevant repositories if required.

Creating a New Security Group

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.

Description			
Inbound			
Outbound			
Tags			
Edit			
Type <small>i</small>	Protocol <small>i</small>	Port Range <small>i</small>	Source <small>i</small>
All traffic	All	All	sg-3e48cf58 (test-doc)
SSH	TCP	22	0.0.0.0/0

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. For information on importing an existing key pair, see [Amazon EC2 Key Pairs](#) in the AWS documentation. If you do not have a key pair, follow these steps:

1. Select **EC2** in **Compute** section of the AWS console.
2. In the **Network & Security** section of 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 the .pem extension) to a secure location and note the location.

You are now ready to [launch an EC2 instance](#).

Launching an EC2 Instance for Cloudera Director

On AWS, Cloudera Director requires a dedicated Amazon EC2 instance. The simplest approach is to create this instance in the same VPC and subnet where you want Cloudera Director to create new instances for Cloudera Manager and your CDH clusters.



Note: Alternatively, you can install Cloudera Director in a different region, on a different cloud provider, or a different network environment. For information on these more complex setups, see [Running Cloudera Director and Cloudera Manager in Different Regions or Clouds](#) on page 79.

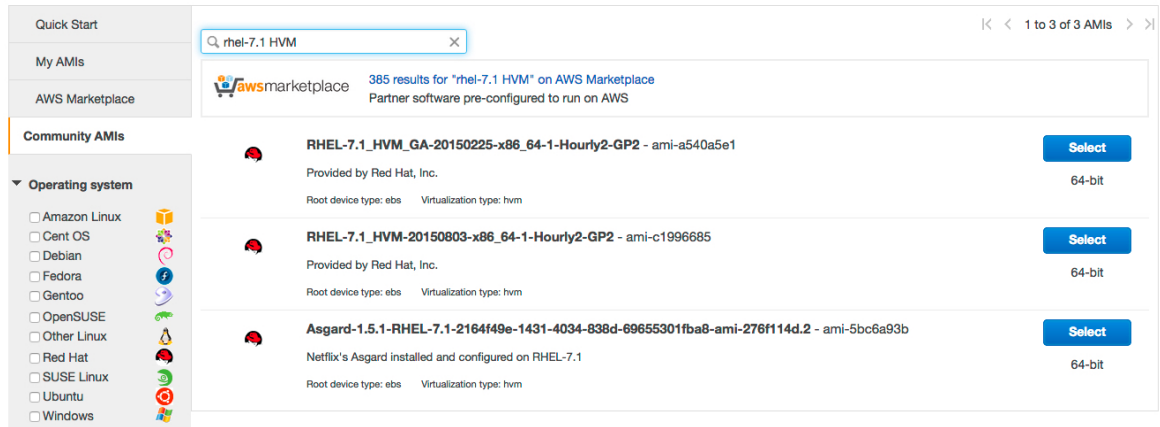
To create the instance, follow these steps:

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 AMI for your Cloudera Director instance. Cloudera recommends that you choose from the Community AMIs list and the latest release of the desired supported distribution. See [Supported Software and Distributions](#) on page 20.
 - a. Select **Community AMIs** in the left pane.
 - b. In the search box, type the desired operating system. For example, if you type `rhel-7.1 HVM`, the search results show the versions of RHEL v7.1 that support HVM. Select the highest GA number to use the latest release of RHEL v7.1 supporting HVM.

Step 1: Choose an Amazon Machine Image (AMI)

[Cancel and Exit](#)

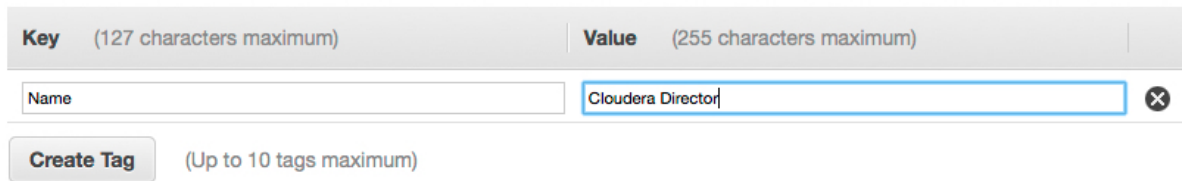
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.



- c. Click **Select** for the AMI version you choose.
- 4. Select the instance type for Cloudera Director. Cloudera recommends using c3.large or c4.large instances.
- 5. Click **Next: Configure Instance Details**.
 - a. Select the correct VPC and subnet.
 - b. The cluster launcher requires Internet access; from the **Auto-assign Public IP** list box, select **Enable**.
 - c. Use the default shutdown behavior, **Stop**.
 - d. Click the **Protect against accidental termination** checkbox.
 - e. (Optional) Click the IAM role drop-down list and select an IAM role.
- 6. Click **Next: Add Storage**. Cloudera Director requires a minimum of 8 GB.
- 7. 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.



- 8. Click **Next: Configure Security Group**.
- 9. On the **Configure Security Group** page, 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 create a new group, enter a **Security group name** and **Description**. To edit an existing group, select the group you want to edit.
 - 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 must be open for the Cloudera Director EC2 instance:

Type	Protocol	Port Range	Source
SSH (22)	TCP (6)	22	0.0.0.0/0

Type	Protocol	Port Range	Source
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:
 - a. Select **Choose an existing key pair** and select the key pair you created in [Setting up the AWS Environment](#) on page 24.
 - b. Click the check box to acknowledge 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.

You are now ready to [install Cloudera Director server and client on the EC2 instance](#).

Installing Cloudera Director Server and Client on the EC2 Instance

To install Cloudera Director, perform the following tasks. You must be either running as root or using sudo to perform these tasks.

RHEL 7 and CentOS 7

1. SSH as `ec2-user` 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.

```
ssh -i your_file.pem ec2-user@private_IP_address
```

Getting Started with Cloudera Director

2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For download and installation information, see [Java SE Downloads](#). After downloading the RPM file to the EC2 instance, install the JDK:

```
sudo yum localinstall jdk-version-linux-x64.rpm
```

3. Some RHEL 7 AMIs do not include `wget` by default. If your RHEL AMI does not, install it now:

```
sudo yum install wget
```

4. Add the Cloudera Director repository to the package manager:

```
cd /etc/yum.repos.d/  
sudo wget "http://archive.cloudera.com/director/redhat/7/x86_64/director/cloudera-director.repo"
```

5. Install Cloudera Director server and client by running the following command:

```
sudo yum install cloudera-director-server cloudera-director-client
```

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

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

7. If the RHEL 7 or CentOS firewall is running on the EC2 instance where you have installed Cloudera Director, disable and stop the firewall with the following commands:

```
sudo systemctl disable firewalld  
sudo systemctl stop firewalld
```

You are now ready to [configure a SOCKS proxy](#).

RHEL 6 and CentOS 6

1. SSH as `ec2-user` 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.

```
ssh -i your_file.pem ec2-user@private_IP_address
```

2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For download and installation information, see [Java SE Downloads](#). After downloading the RPM file to the EC2 instance, install the JDK:

```
sudo yum localinstall jdk-version-linux-x64.rpm
```

3. Add the Cloudera Director repository to the package manager:

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

4. Install Cloudera Director server and client by running the following command:

```
sudo yum install cloudera-director-server cloudera-director-client
```

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

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

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

```
sudo service iptables save
sudo chkconfig iptables off
sudo service iptables stop
```

You are now ready to [configure a SOCKS proxy](#).

Ubuntu

1. SSH as `ubuntu` 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.

```
ssh -i your_file.pem ubuntu@private_IP_address
```

2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For download and installation information, see [Java SE Downloads](#). After downloading the installation file to the EC2 instance, install the JDK. The following example installs JDK version 7:

```
sudo apt-get update
sudo apt-get install oracle-j2sdk1.7
```

3. Add the Cloudera Director repository to the package manager:

```
cd /etc/apt/sources.list.d/
sudo curl "http://archive.cloudera.com/director/ubuntu/trusty/amd64/director/cloudera-director.list" -O
```

4. Add the signing key:

```
sudo 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 following command:

```
sudo apt-get update
sudo apt-get install cloudera-director-server cloudera-director-client
```

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

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

7. Save the existing firewall rules and disable the firewall:

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

You are now ready to [configure a SOCKS proxy](#).

Installing Only Cloudera Director Server or Cloudera Director Client

The installation instructions above will install both the server and client. Cloudera recommends installing both because together they provide the full functionality of Cloudera Director. Optionally, you can install just the client, but this will only enable you to use the client in standalone mode. Similarly, you can install just the server, but then you will be unable to launch a cluster at the command line with a customized configuration file. For more information on the Cloudera Director client and server, and how they work together, see [Cloudera Director Client and Server](#) on page 9.

To install only Cloudera Director client, run one of the following installation commands in place of the command given above:

- For RHEL and CentoOS, run the command `sudo yum install cloudera-director-client` instead of `sudo yum install cloudera-director-server cloudera-director-client`.

Getting Started with Cloudera Director

- For Ubuntu: run the command `sudo apt-get install cloudera-director-client` instead of `sudo apt-get install cloudera-director-server cloudera-director-client`.

To install only Cloudera Director server, run one of the following installation commands in place of the command given above:

- For RHEL and CentoOS, run the command `sudo yum install cloudera-director-server` instead of `sudo yum install cloudera-director-server cloudera-director-client`.
- For Ubuntu: run the command `sudo apt-get install cloudera-director-server` instead of `sudo apt-get install cloudera-director-server cloudera-director-client`.

Configuring a SOCKS Proxy for Amazon EC2

In AWS, the security group that you create and specify for your EC2 instances functions as a firewall to prevent unwanted access to your cluster and Cloudera Manager. For security purposes, Cloudera recommends that you do not configure security groups to allow internet access to your instances on their public IP addresses. Instead, Cloudera recommends that you connect to your cluster and to Cloudera Manager using a [SOCKS proxy server](#). A SOCKS proxy server allows a client (such as your web browser) to connect directly and securely to a server (such as your Cloudera Director server web UI) and, from there, to the web UIs on other IP addresses and ports in the same subnet, including the Cloudera Manager and HUE web UIs. So, the SOCKS proxy provides access to the Cloudera Director UI, Cloudera Manager UI, HUE UI, and any other cluster web UIs without exposing those ports outside the subnet.



Note: The same result could be achieved by configuring an SSH tunnel from your browser to the EC2 instance. But an SSH tunnel enables traffic from a single client (IP address and port) to a single server (IP address and port), so this approach would require you to configure a separate SSH tunnel for each connection.

To set up a SOCKS proxy for your web browser, follow the steps below.

Step 1: Set Up a SOCKS Proxy Server with SSH

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

```
nohup ssh -i "your-key-file.pem" -CND 8157 ec2-user@instance_running_director_server &
```

where

- `nohup` (optional) is a POSIX command to ignore the HUP (hangup) signal so that the proxy process is not terminated automatically if the command process is later terminated.
- `C` sets up compression.
- `N` suppresses any command execution once established.
- `D 8157` sets up the SOCKS 5 proxy on the port. (The port number 8157 in this example is arbitrary, but must match the port number you specify in your browser configuration in the next step.)
- `&` (optional) causes the SSH connection to run as an operating system background process, independent of the command shell. (Without the `&`, you would leave your terminal open while the proxy server is running and use another terminal window to issue other commands.)



Important: If you are using a PAC file, the port specified in the PAC file must match the port used in the ssh command (port 8157 in the example above).

You are now ready to [deploy Cloudera Manager and CDH](#).

Step 2: Configure Your Browser to Use the Proxy

Next, configure your browser settings to use the SOCKS proxy.

On Google Chrome

By default, Google Chrome uses system-wide proxy settings on a per-profile basis. To get around that you can launch Chrome via the command line and specify the following:

- The SOCKS proxy port to use (this must be the same value used above)
- The profile to use (this example will create a new profile)

This will create a new profile and launch a new instance of Chrome that won't interfere with your current running instance of Chrome.

Linux

```
/usr/bin/google-chrome \
--user-data-dir="$HOME/chrome-with-proxy" \
--proxy-server="socks5://localhost:8157"
```

Mac OS X

```
"/Applications/Google Chrome.app/Contents/MacOS/Google Chrome" \
--user-data-dir="$HOME/chrome-with-proxy" \
--proxy-server="socks5://localhost:8157"
```

Microsoft Windows

```
"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" ^
--user-data-dir="%USERPROFILE%\chrome-with-proxy" ^
--proxy-server="socks5://localhost:8157"
```

Now in this Chrome session you can connect to any Cloudera Director accessible host using the private IP address or internal FQDN. For example, if you proxy to the Cloudera Director server, you can connect to Cloudera Director as if it were local by putting `localhost:7189` in Chrome's URL bar.

Setting Up SwitchyOmega on the Google Chrome Browser

If you are using Google Chrome, and especially if you use multiple proxies, the SwitchyOmega browser extension is a convenient tool for configuring and managing all of your proxies in one place and for switching from one proxy to another.

1. Open Google Chrome and go to [Chrome Extensions](#).
2. Search for **Proxy SwitchyOmega** and add to it Chrome.
3. In the **Profiles** menu of the **SwitchyOmega Options** screen, click **New profile** and do the following:
 - a. In the **Profile Name** field, enter `AWS-Cloudera`.
 - b. Select the type **PAC Profile**.
 - c. The [proxy autoconfig](#) (PAC) script contains the rules required for Cloudera Director. Enter or copy the following into the **PAC Script** field:

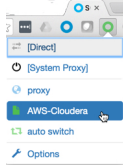
```
function regExpMatch(url, pattern) {
  try { return new RegExp(pattern).test(url); } catch(ex) { return false; }
}

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';
}
```

4. In the **Actions** menu, click **Apply Changes**.

5. On the Chrome toolbar, select the **AWS-Cloudera** profile for SwitchyOmega.



Deploying Cloudera Manager and CDH on AWS

To deploy Cloudera Manager and CDH on an AWS EC2 instance, 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.



Note: The lifecycle of instances and clusters depends on the availability of external repositories (for example, the Cloudera Manager repository). If these repositories are unreachable during this lifecycle, Cloudera Director cannot grow the cluster, and a grow operation results in a `Modify failed` state until the repository is available again. To ensure that there is no point of failure during cluster growth, you can preload the AMIs you use with Cloudera Manager and CDH.

To create an environment:

1. Open a web browser and go to the private IP address of the instance you created in [Launching an EC2 Instance for Cloudera Director](#) on page 25. 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, Cloudera Manager, and a CDH cluster.

4. In the **Add Environment** screen:
 - a. Enter a name in the **Environment Name** field.
 - b. Select **Amazon Web Services (AWS)** from the **Cloud provider** field.
 - c. Enter your AWS credentials in the **Access key ID** and **Secret access key** fields.
 - d. In the **EC2 region** field, select the same region in which your Cloudera Director instance was created.

Add Environment

GENERAL INFORMATION

Environment name * ?

Cloud provider ?

Access key ID ?

Secret access key ?

EC2 (ELASTIC CLOUD COMPUTE)

EC2 region ?

> Advanced Options

e. In the **SSH Credentials** section:

- Enter **ec2-user** in the **Username** field.
- Copy the SSH private key you created in [Launching an EC2 Instance for Cloudera Director](#) on page 25 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, click **Select a Template** if you already have one that you want to use, otherwise, click **Create New Instance Template**.

The **Create New Instance Template** modal screen displays.

7. In the **Create New Instance Template** modal screen:

- 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 [Launching an EC2 Instance for Cloudera Director](#) on page 25, or find another AMI with a supported operating system.

- d. In the **Tags** field, add one or more tags to associate with the instance.
- e. In the **Security group IDs** field, enter the security group ID you set up in [Creating a New Security Group](#) on page 24.
- f. In the **VPC subnet ID** field, enter the ID of the VPC subnet that was created during [VPC setup](#).
- g. 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 **Desired License Type** field, select one of the following license types:

- Cloudera Enterprise: includes the core CDH services (HDFS, Hive, Hue, MapReduce, Oozie, Sqoop, YARN, and ZooKeeper) and, depending on the license edition, one or more additional services (Accumulo, HBase, Impala, Navigator, Solr, Spark). For more information on Cloudera Enterprise licenses, see [Managing Licenses](#) in the Cloudera Manager documentation.
- Cloudera Enterprise Trial: a 60-day trial license that includes all CDH services.
- Cloudera Express: no license required.

Licensing

Desired License Type * ?

Please provide a Cloudera Manager license key.

License Key * File Upload Direct Input ?

Choose File

Billing ID ?

To enable usage-based billing, you must have a Cloudera Enterprise license and a billing ID provided by Cloudera. Perform these steps in the **Add Cloudera Manager** screen:

1. In the **Desired License Type** field, select **Cloudera Enterprise**.

2. In the **License Key** field, either select a Cloudera Enterprise license file to upload or select **Direct Input** and input the license file text directly into the text area.
 3. To enable usage-based billing, enter the billing ID provided to you by Cloudera in the **Billing ID** field.
9. In the **Add Cloudera Manager** screen, click **Cloudera Manager Configurations**.
 10. In the **Cloudera Manager Configurations** modal screen, set the heap size:
 - a. In the **Scope** field, select **Host Monitor** and add `firehose_heapsize` and `1073741824` in the respective **Name** and **Value** fields.
 - b. Click **+**.
 - c. In the **Scope** field, select **Service Monitor** and add `firehose_heapsize` and `1073741824` in the respective **Name** and **Value** fields.
 - d. Click **Save Changes**.

Cloudera Manager Configurations ×

Scope: Service Monitor (modified) ?

firehose_heapsize: - +

[- Hide All Configurations](#)

Configuration	Value	Scope
firehose_heapsize	1073741824	Host Monitor
firehose_heapsize	1073741824	Service Monitor

Cancel Reset **Save Changes**

11. By default, the version of Cloudera Manager installed depends on the version of Cloudera Director you are using:
 - If you are using Cloudera Director 2.0, the latest released version of Cloudera Manager 5.5 is installed by default.
 - If you are using Cloudera Director 2.1, the latest released version of Cloudera Manager 5.7 is installed by default.

To install a version of Cloudera Manager higher or lower than the default version, perform the following steps:

- a. In the **Configurations** section, check **Override default Cloudera Manager repository**.
- b. In the **Repository URL** field, enter the repository URL for the version of Cloudera Manager to install. Repository URLs for versions of Cloudera Manager 5 have the form <http://archive.cloudera.com/cm5/> followed by the operating system, operating system major version, processor architecture, cm (for Cloudera Manager), and the Cloudera Manager major, minor, and (if applicable) maintenance release number. For example, for Cloudera Manager 5.5.4, the repository URL is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.5.4/.



Note: The Cloudera Manager minor version must be the same as or higher than the CDH minor version. For example, Cloudera Manager 5.5 cannot be used to launch or manage a CDH 5.7 cluster, but Cloudera Manager 5.7 can be used with a CDH 5.7 (or lower) cluster.

- c. In the **Repository Key URL** field, enter the URL for the repository key. Repository key URLs have the same form as repository URLs except they end with the name of the key file instead of the Cloudera Manager

version. For example, the repository key URL for any version of Cloudera Manager 5 on any supported version of Red Hat 7 is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera.

12 In the **Add Cloudera Manager** screen, click **Continue**.


13 At the **Confirmation** prompt, click **OK** to begin adding a cluster.

14 On the **Add Cluster** screen:

- a. Enter a name for the cluster in the **Cluster name** field.
- b. Enter the version of CDH to deploy in the **Version** field or leave the default value. By default, the version of CDH installed depends on the version of Cloudera Director you are using:
 - If you are using Cloudera Director 2.0, the latest released version of CDH 5.5 is installed by default.
 - If you are using Cloudera Director 2.1, the latest released version of CDH 5.7 is installed by default.

To install a version of CDH higher or lower than the default version, perform the following steps:

- a. Enter the desired CDH version in the **Version** field of the **Products** section. For example, for CDH 5.4.8 enter 5.4.8.
- b. Scroll down to **Configurations (optional)** and expand the section.
- c. Click **Override default parcel repositories**.
- d. Enter the repository parcel URL for the version of CDH you want to install. Parcel URLs for versions of CDH 5 have the form <http://archive.cloudera.com/cdh5/parcels/>, followed by the major, minor, and (if applicable) maintenance release number. For example, the URL for CDH 5.4.8 is <http://archive.cloudera.com/cdh5/parcels/5.4.8>.

 **Note:** The CDH minor version must not be higher than the Cloudera Manager minor version. For example, CDH 5.7 will not work with Cloudera Manager 5.5, but CDH 5.7 (or lower) will work with Cloudera Manager 5.7.

- c. In the **Services** section, select the services you want to install.
- d. In the **Instance groups** area, create a new template for the groups or for each group and the number of instances you want. If you want to use Spot instances for your **workers** group:
 - a. In the **Create New Instance Template** modal screen, click **Advanced Options**.
 - b. In the **Spot bid (USD/hr)** field, enter your Spot bid price.
 - c. Click the **Use Spot instances** checkbox.
 - d. Click **Save Changes**.

For more information about using Spot instances with Cloudera Director, see [Using Spot Instances](#) on page 77.

Instance groups					
Name [?]	Roles	Instance Template	Instance Count		
masters	Edit Roles	TEST-TEMPLATE Edit	1 ↕	Delete Group	
workers	Edit Roles	TEST-TEMPLATE Edit	5 ↕	Delete Group	
gateway	Edit Roles	TEST-TEMPLATE Edit	1 ↕	Delete Group	
Add Group					

15 Click **Continue**.

16 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

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

You are finished with the deployment tasks.

Cleaning Up Your AWS Deployment

When you are done testing or using Cloudera Director, terminate your instances to stop incurring charges to your AWS account.

1. In Cloudera Director, terminate each instance in your clusters.
 - a. Click an environment name.
 - b. In the **Actions** column, select **Terminate Cluster**.
 - c. Repeat for each environment you configured.
2. If you want to save anything in Cloudera Director (the configuration file or database, for example), back it up.
3. In the AWS Management Console, terminate the Cloudera Director instance and any other instance Cloudera Director was unable to terminate.
4. If applicable, terminate any external database you configured Cloudera Director to use.

Getting Started on Google Cloud Platform

To use Cloudera Director on Google Cloud Platform, you create a project, start an instance in Google Compute to run Cloudera Director, and create a secure connection. This section details steps for each of these tasks.



Important: Cloudera Director supports preemptible virtual machines. Preemptible virtual machines are short-lived instances that have a lower cost but are subject to reclamation at any time by Google Compute Engine. Because of the possibility of interruption, we recommend that you use preemptible virtual machines only for worker roles in a cluster, not for master or gateway roles. For more information, see the Google Cloud Platform's [Preemptible Virtual Machines](#) page.

Creating a Google Cloud Platform Project

To run Cloudera Director on Google Cloud Platform, begin by creating a project:

1. Go to the [Google Cloud Platform](#) web site.
2. Click **My console** in the upper-right corner of the screen.
3. Select your Google account, and sign in.

Your screen is redirected to the **Google Developers Console**.

4. In the **Google Developers Console**, click **Select a project > Create a project**.
5. In the **New Project** form, enter a project name, click that you agree to the terms of service, and click **Create**.



Note: To create a project in Google Cloud Platform, first create a billing account or a free trial account, or sign into an existing billing account. To create an account, click **Create new billing account** in the Google Developers Console.

You are ready to [configure tools](#) for your project.

Configuring Tools for Your Google Cloud Platform Account

Before installing Cloudera Director, Cloudera recommends that you configure some tools for your Google Cloud Platform account.

1. Create a service account for Cloudera Director.
2. Create an SSH key.
3. Set up gcloud compute.

Creating a Service Account for Cloudera Director

A service account enables Cloudera Director to authenticate to various Google Cloud Platform services, such as Google Cloud Storage. To create a service account, perform the following steps:

1. Ensure that the Google Compute Engine API is enabled. In the Google Cloud Platform console for your project, click **API Manager**.
2. Click **Compute Engine API** (under **Google Cloud APIs**).
3. If not already enabled, click **Enable API**.
4. At the prompt, click **Enable Billing**.
5. At the prompt, select the billing account and click **Set account**.

A status displays, showing that the Google Compute Engine API is enabling.



Google Compute Engine

Google Compute Engine provides virtual machines for large scale data processing and analytics applications.

[Learn more](#)

[Try this API in APIs Explorer](#)

6. Click **API Manager**.
7. In the **API Manager** menu, click **Credentials**.
8. In the **Credentials** screen, click **New credentials** > **Service account key**.
9. In the **Create service account key** screen, click **JSON** and click **Create**.



Create service account

Key type

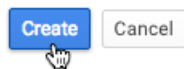
Downloads a file that contains the public/private key pair. It is the only copy of the key, so store it securely.

JSON

Recommended

P12

For backward compatibility with code using the P12 format



You are prompted to save the JSON file to your local machine. Note the location where you download this file. You will be prompted to select this file later, when you create an environment in Cloudera Director.

Creating and Uploading an SSH Key

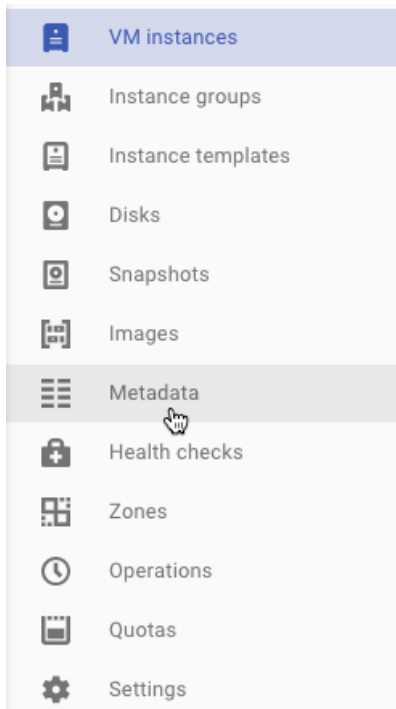
To SSH into an instance using your own terminal (as opposed to the Google Cloud Platform console), you must generate and upload an SSH key.

1. Generate an SSH key using the following command:

```
$ ssh-keygen -f ~/.ssh/my_gcp_keyname -t rsa
```

This generates a public/private key pair.

2. In the **Compute Engine** menu, click **Metadata**.



3. Click the **SSH Keys** tab and click **Add SSH Keys**.
4. Copy your key data into the input box in the following format:

```
protocol public-key-data username@example.com
```

5. Click **Save**. Your public key is now available to all instances in the project.

Installing gcloud compute

Cloudera recommends installing the `gcloud compute` command-line tool because it allows you to manage your Google Compute Engine resources more easily. To install and configure `gcloud compute`, follow the instructions at [gcloud compute](#).

You are ready to [create a new VM instance](#) within your project.

Creating a Google Compute Engine VM Instance

Once you have created or selected a project in the Google Developers Console, you can create a new VM instance in your project.

1. In the left side menu of the Google Developers Console, click **Compute > Compute Engine > VM instances**.
2. Click **Create Instance**.
3. Provide the following values to define your VM instance:

Table 1: VM Instance Values

Name	Description	Details/Restrictions
Name	Name of the instance.	The name must start with a lowercase letter followed by up to 62 lowercase letters, numbers, or hyphens. The name cannot end with a hyphen.
Zone	Where your data is stored.	Some resources can only be used by other resources in the same zone or region. For example, to attach a disk to a VM instance, both resources must reside in the same zone. For more information, see Regions and Zones in the Google GPC documentation.
Machine type	The number of CPUs and amount of memory for your instance.	Cloudera recommends a machine type of at least n1-standard-1 for this Quick Start instance. For a production instance, Cloudera recommends at least an n1-standard-2 instance for running Cloudera Director and an n1-highmem-8 instance for running Cloudera Manager and CDH.
Boot disk	The disk to boot from.	Select a preconfigured image with a version of Linux supported for Cloudera Director. For more information about supported Linux versions, see Supported Software and Distributions on page 20.
Boot disk type	The type of boot disk.	For this Quick Start, choose standard persistent disk for less expensive storage space. A solid-state persistent disk (SSD) is better suited to handling high rates of random I/O operations per second (IOPS) or streaming throughput with low latency.
Firewall	Traffic to block.	Leave both HTTP and HTTPS traffic unchecked.
Project access	Access to Google Cloud services.	Leave this unchecked (disabled). These services are not used in this QuickStart.
Management, disk, networking, access & security options	Additional options available when you click the double arrows.	Use the default values for all of these settings.

You are now read to [install Cloudera Director Server and Client](#) on your instance.

Installing Cloudera Director Server and Client on Google Compute Engine

Cloudera recommends that you install Cloudera Director server on your cloud provider in the subnet where you will create CDH clusters, because Cloudera Director must have access to the private IP addresses of the instances that it creates. To install Cloudera Director server, perform the following tasks.



Note: You must be either running as root or using sudo to perform these tasks.

RHEL 6 and CentOS 6

1. In the **Compute Engine > VM instances** screen, click the **SSH** link next to your instance name.

This opens a new window.



Note: Alternatively, you can connect to your instance using:

- SSH in a terminal using the following command:

```
ssh -i your_key_file -o UserKnownHostsFile=/dev/null \
-o CheckHostIP=no -o StrictHostKeyChecking=no user@ip_address
```

- The `gcloud compute ssh` command. When you connect to your instance for the first time using the `gcloud compute` command-line tool, `gcloud` automatically creates an SSH key and inserts it into the instance.

2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For installation information, see [Java SE Downloads](#).

3. Download Cloudera Director by running the following commands:

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

4. Install Cloudera Director server by running the following command:

```
sudo yum install cloudera-director-server cloudera-director-client
```

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

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

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

```
sudo service iptables save
sudo chkconfig iptables off
sudo service iptables stop
```

You are now ready to [configure a SOCKS proxy](#) for your instances.

RHEL 7 and CentOS 7

1. In the **Compute Engine > VM instances** screen, click the **SSH** link next to your instance name.

This opens a new window.



Note: Alternatively, you can connect to your instance using:

- SSH in a terminal using the following command:

```
ssh -i your_key_file -o UserKnownHostsFile=/dev/null \
-o CheckHostIP=no -o StrictHostKeyChecking=no user@ip_address
```

- The `gcloud compute ssh` command. When you connect to your instance for the first time using the `gcloud compute` command-line tool, `gcloud` automatically creates an ssh key and inserts it into the instance.

2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For installation information, see [Java SE Downloads](#).

3. Download Cloudera Director by running the following commands:

```
cd /etc/yum.repos.d/
sudo wget "http://archive.cloudera.com/director/redhat/7/x86_64/director/cloudera-director.repo"
```

4. Install Cloudera Director server by running the following command:

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

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

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

6. Disable and stop the firewall with the following commands:

```
sudo systemctl disable firewalld
sudo systemctl stop firewalld
```

You are now ready to [configure a SOCKS proxy](#) for your instances.

Ubuntu

1. In the **Compute Engine > VM instances** screen, click the **SSH** link next to your instance name.

This opens a new window.



Note: Alternatively, you can connect to your instance using:

- SSH in a terminal using the following command:

```
ssh -i your_key_file -o UserKnownHostsFile=/dev/null \
-o CheckHostIP=no -o StrictHostKeyChecking=no user@ip_address
```

- The `gcloud compute ssh` command. When you connect to your instance for the first time using the `gcloud compute` command-line tool, `gcloud` automatically creates an SSH key and inserts it into the instance.

2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For installation information, see [Java SE Downloads](#).
3. Download Cloudera Director by running the following commands:

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

4. Add the signing key by running 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 following command:

```
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:

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

7. Save the existing firewall rules and disable the firewall:

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

You are now ready to [configure a SOCKS proxy](#) for your instances.

Configuring a SOCKS Proxy for Google Compute Engine

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

To set up a SOCKS proxy, follow the steps in the Google Compute Engine documentation, [Securely Connecting to VM Instances](#), and follow the instructions for setting up a SOCKS proxy over SSH.

Once you have set up a SOCKS proxy, you can [deploy Cloudera Manager and CDH](#).

Deploying Cloudera Manager and CDH on Google Compute Engine

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

To create an environment:

1. Open a web browser and go to the private IP address of the instance you created in [Creating a Google Compute Engine VM Instance](#) on page 39. 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:
 - a. Enter a name in the **Environment Name** field.
 - b. In the **Cloud provider** field, select **Google Cloud Provider**.
 - c. In the **Project ID** field, enter the ID for the project you created in [Creating a Google Cloud Platform Project](#) on page 37.
 - d. In the **Advanced Options** area, upload or copy the JSON key to the **Client ID JSON Key** field. You created this key in [Configuring Tools for Your Google Cloud Platform Account](#) on page 38.

Add Environment

GENERAL INFORMATION

Environment name * ?

Cloud provider ?

Project ID * ?

▼ **Advanced Options**

Client ID JSON Key File Upload Direct Input ?

```
e/iN4x1KlAjNCDXskL+tjyn6uchSs
\neyJ6PlJpGHANDPTMvuJSzsk\u003d
\n-----END PRIVATE KEY-----\n",
  "client_email": "426487107067-
bq66q975p5g5e1a9815t4ki4tvqbjquc@deve
loper.gserviceaccount.com",
  "client_id": "426487107067-
bq66q975p5g5e1a9815t4ki4tvqbjquc.apps
.googleusercontent.com",
  "type": "service_account"}
```

- e. In the **Advanced Options** section, enter the same **region** that your Cloudera Director instance was created in.
- f. In the **SSH Credentials** section:
 - Enter a username in the **Username** field. Google Compute will create the user specified here.
 - Copy the SSH private key you created in [Creating and Uploading an SSH Key](#) on page 39 in the **Private key** field.

GOOGLE COMPUTE ENGINE

▼ **Advanced Options**

Region ?

SSH CREDENTIALS

Username * ?

Private key * File Upload Direct Input ?

```

0M57r/p5
u89wUtSzk7
/vX1xNhp9sQiuTzs6KtYTSNrK9GwdQ2fqBAoG
Af0mVgn4WgKZ6TamDrifBL4ocAaBx
ih36YNghy736AxL3AHQ19Mna14QA
/2d5Q0CQ031
/MdssSufqhSeMA1ht0IZpdz3xNrBsms84G4Y1
40gAqiKV0QMUYkKBB9tgnLd175m58xDHEe0UM
yyqGm+AryQPW35B1Ak4CMWEeWTQYDo=
-----END RSA PRIVATE KEY-----

```

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 ?

Select a Template

Create New 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 **n1-highmem-4** or **n1-highmem-8**.
- In the **Machine type** field, enter the machine type you chose in [Creating a Google Compute Engine VM Instance](#) on page 39.

- In the **Tags** field, add one or more tags to associate with the instance.
- Click **Save changes**.

8. In the **Add Cloudera Manager** screen, click **Cloudera Manager Configurations**.

The **Cloudera Manager Configurations** modal screen displays.

9. In the **Cloudera Manager Configurations** modal screen, set the heap size:

- In the **Scope** field, select **Host Monitor** and add `firehose_heapsize` and `1073741824` in the respective **Name** and **Value** fields.
- Click **+**.
- In the **Scope** field, select **Service Monitor** and add `firehose_heapsize` and `1073741824` in the respective **Name** and **Value** fields.
- Click **Save Changes**.

The screenshot shows the 'Cloudera Manager Configurations' modal. At the top, the 'Scope' is set to 'Service Monitor (modified)'. Below this, there is a configuration entry for 'firehose_heapsize' with a value of '1073741824'. A table below lists two configurations:

Configuration	Value	Scope
firehose_heapsize	1073741824	Host Monitor
firehose_heapsize	1073741824	Service Monitor

At the bottom of the modal, there are three buttons: 'Cancel', 'Reset', and 'Save Changes'.

10. By default, the version of Cloudera Manager installed depends on the version of Cloudera Director you are using:

- If you are using Cloudera Director 2.0, the latest released version of Cloudera Manager 5.5 is installed by default.
- If you are using Cloudera Director 2.1, the latest released version of Cloudera Manager 5.7 is installed by default.

To install a version of Cloudera Manager higher or lower than the default version, perform the following steps:

- In the **Configurations** section, check **Override default Cloudera Manager repository**.
- In the **Repository URL** field, enter the repository URL for the version of Cloudera Manager to install. Repository URLs for versions of Cloudera Manager 5 have the form <http://archive.cloudera.com/cm5/> followed by the operating system, operating system major version, processor architecture, cm (for Cloudera Manager), and the Cloudera Manager major, minor, and (if applicable) maintenance release number. For example, for Cloudera Manager 5.5.4, the repository URL is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.5.4/.



Note: The Cloudera Manager minor version must be the same as or higher than the CDH minor version. For example, Cloudera Manager 5.5 cannot be used to launch or manage a CDH 5.7 cluster, but Cloudera Manager 5.7 can be used with a CDH 5.7 (or lower) cluster.

- In the **Repository Key URL** field, enter the URL for the repository key. Repository key URLs have the same form as repository URLs except they end with the name of the key file instead of the Cloudera Manager

version. For example, the repository key URL for any version of Cloudera Manager 5 on any supported version of Red Hat 7 is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera.

11 In the **Add Cloudera Manager** screen, click **Continue**.

12 At the **Confirmation** prompt, click **OK** to begin adding a cluster.

13 On the **Add Cluster** screen:

- Enter a name for the cluster in the **Cluster name** field.
- Enter the version of CDH to deploy in the **Version** field or leave the default value. By default, the version of CDH installed depends on the version of Cloudera Director you are using:
 - If you are using Cloudera Director 2.0, the latest released version of CDH 5.5 is installed by default.
 - If you are using Cloudera Director 2.1, the latest released version of CDH 5.7 is installed by default.

To install a version of CDH higher or lower than the default version, perform the following steps:

1. Enter the desired CDH version in the **Version** field of the **Products** section. For example, for CDH 5.4.8 enter 5.4.8.
2. Scroll down to **Configurations (optional)** and expand the section.
3. Click **Override default parcel repositories**.
4. Enter the repository parcel URL for the version of CDH you want to install. Parcel URLs for versions of CDH 5 have the form <http://archive.cloudera.com/cdh5/parcels/>, followed by the major, minor, and (if applicable) dot release number. For example, the URL for CDH 5.4.8 is <http://archive.cloudera.com/cdh5/parcels/5.4.8>.



Note: The CDH minor version must not be higher than the Cloudera Manager minor version. For example, CDH 5.7 will not work with Cloudera Manager 5.5, but CDH 5.7 (or lower) will work with Cloudera Manager 5.7.

- In the **Services** section, select the services you want to install.
- In the **Instance groups** area, create a new template for the groups or for each group and the number of instances you want.

Instance groups					
Name	Roles	Instance Template		Instance Count	
masters	Edit Roles	TEST-TEMPLATE	Edit	1	Delete Group
workers	Edit Roles	TEST-TEMPLATE	Edit	5	Delete Group
gateway	Edit Roles	TEST-TEMPLATE	Edit	1	Delete Group
Add Group					

14 Click **Continue**.

15 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

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

You are finished with the deployment tasks.

Cleaning Up Your Google Cloud Platform Deployment

When you are done testing or using Cloudera Director, terminate your instances to stop incurring charges to your Google Cloud Platform account.

1. In Cloudera Director, terminate each instance in your clusters.
 - Click an environment name.
 - In the **Actions** column, select **Terminate Cluster**.
 - Repeat for each environment you configured.
2. If you want to save anything in Cloudera Director (the configuration file or database, for example), back it up.
3. In the Google Compute Console, delete the Cloudera Director instance and any other instance Cloudera Director was unable to delete.
4. If applicable, delete any external database you configured Cloudera Director to use.

Getting Started on Microsoft Azure

Before you can use Cloudera Director to deploy a cluster on Microsoft Azure, you must create the Azure resources the cluster requires. This section describes the resources you must create and steps on how to create them.

For best practices when creating a cluster on Microsoft Azure, refer to the [Cloudera Enterprise Reference Architecture for Azure Deployments](#).

Obtaining Credentials for Cloudera Director

Create an [Active Directory \(AD\) application and service principal](#). The service principal is tied to the AD application, and Cloudera Director uses the service principal credentials to create and delete resources on Microsoft Azure. Therefore, you must make sure the AD application has the **contributor** role in your Azure subscription, which allows permission to create and delete resources. If you are not sure about these settings, contact your Active Directory administrator or Microsoft Azure Support.

The service principal is typically created by a system administrator or security administrator of your organization. This person must have administrator privileges for your Microsoft Azure subscription.

Once the Azure service principal is created, obtain the following four kinds of Azure credentials for Cloudera Director:

- Subscription ID
- Tenant ID
- Client ID
- Client Secret

You can get the subscription ID in the Azure Portal (either the new or old portal); see the [Azure subscriptions blade](#).

You can create the AD application and service principal, get the tenant ID, client ID, and client secret, and assign the contributor role to the newly-created AD application by following one of these two methods:

1. The [Azure Portal Steps](#) (this method is recommended, as it is easier to follow)
2. The [Azure CLI Steps](#)



Note: The *client secret* is referred to as the application *password* in the [Azure CLI Steps](#) documentation.

If you are having trouble finding this information, contact Microsoft support.

Setting up Azure Resources

This section describes the setup of various resources required by Microsoft Azure:

Setting Up Resource Groups

Set up resource groups for the Azure resources required by Cloudera Director:

- For housing a cluster's Azure virtual machines (VMs)
- For housing a cluster's Azure virtual network (VNet)
- For housing a cluster's Azure network security group (NSG)

The resources above typically have different lifecycles, so you may want to place each in a separate resource group for convenience. For simplicity, you can also place them under the same resource group instead.

Creating a New Resource Group

To create a new resource group, perform the following steps:

1. In the left pane, click **New**.
2. Type `resource group` in the search box.
3. Click **Resource Group** in the search result.
4. Click **Create**.
5. Type in a name for the resource group.

Repeat these steps to create multiple resource groups for Azure resources.

Setting Up a Network Security Group

This section explains how to create a new network security group (NSG) in Azure.

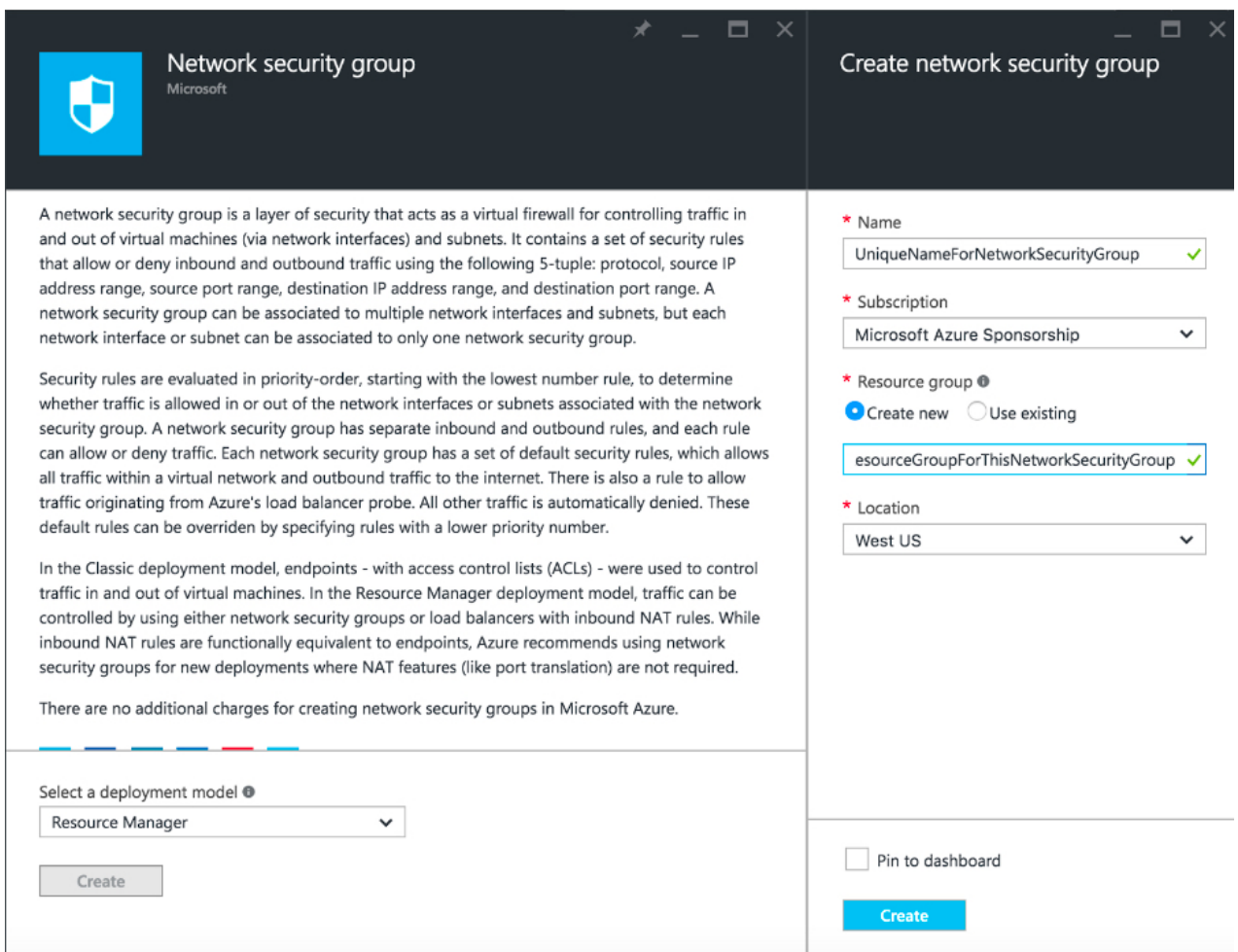
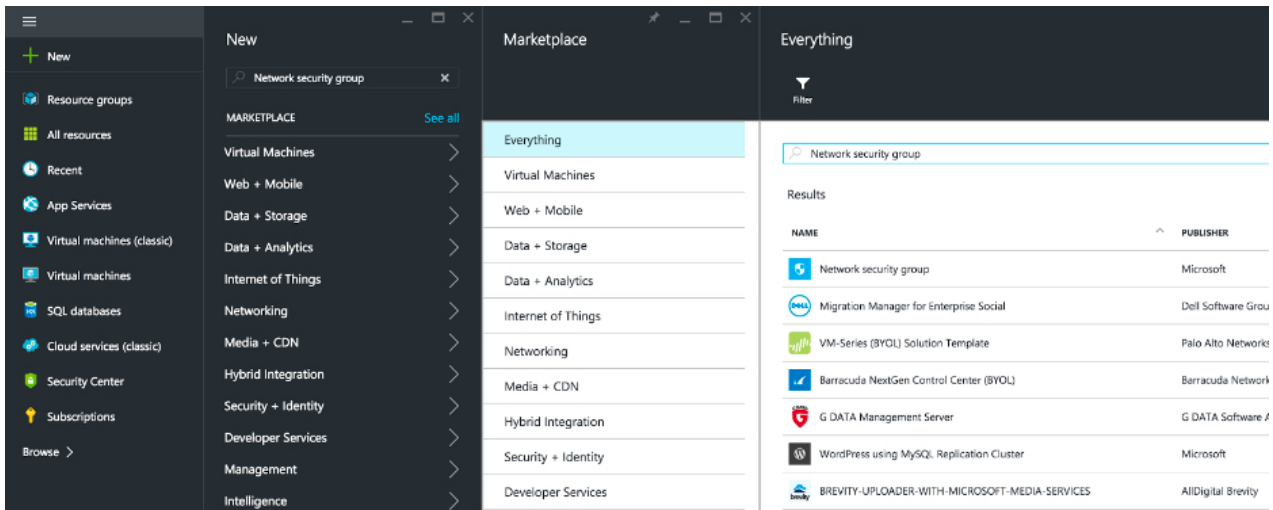


Note: Cloudera Director requires port 7189 to be opened if you want to allow access to the Cloudera Director web UI from public IP addresses. Cloudera Manager requires port 7180 to be opened if you want to allow access to the Cloudera Manager web UI from public IP addresses.

Creating a New Network Security Group

To create a new network security group:

1. In the left pane, click **New**.
2. Type `Network security group` in the search box.
3. Click **Network security group** in the search result.
4. Click **Create**.
5. Type in a name for the network security group.
6. Type in a name for new resource group or select an existing resource group.
7. Click **Create**.
8. Once created, see [How to manage NSGs using the Azure portal](#) in the Microsoft Azure documentation for instructions on creating the rules in the Network security group



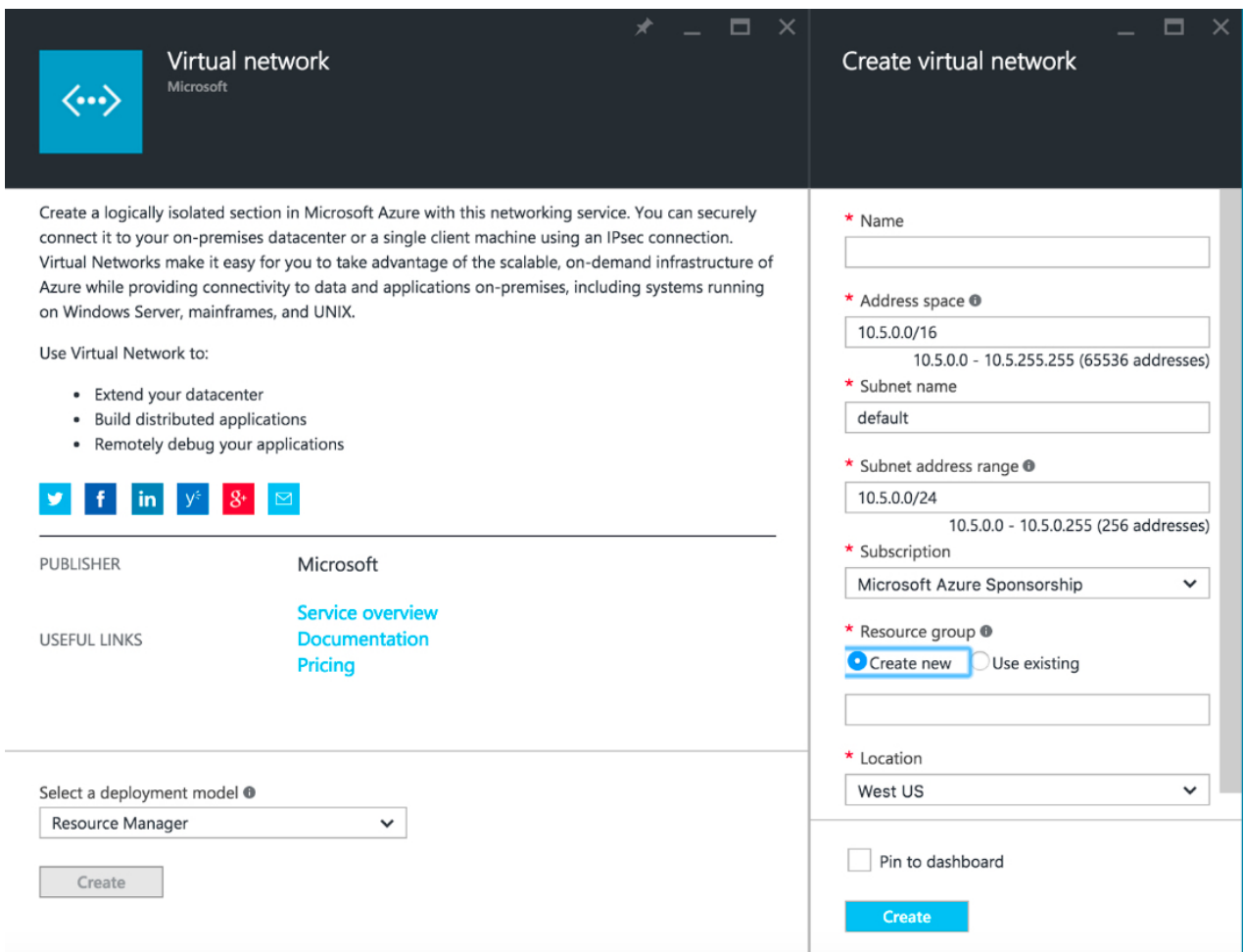
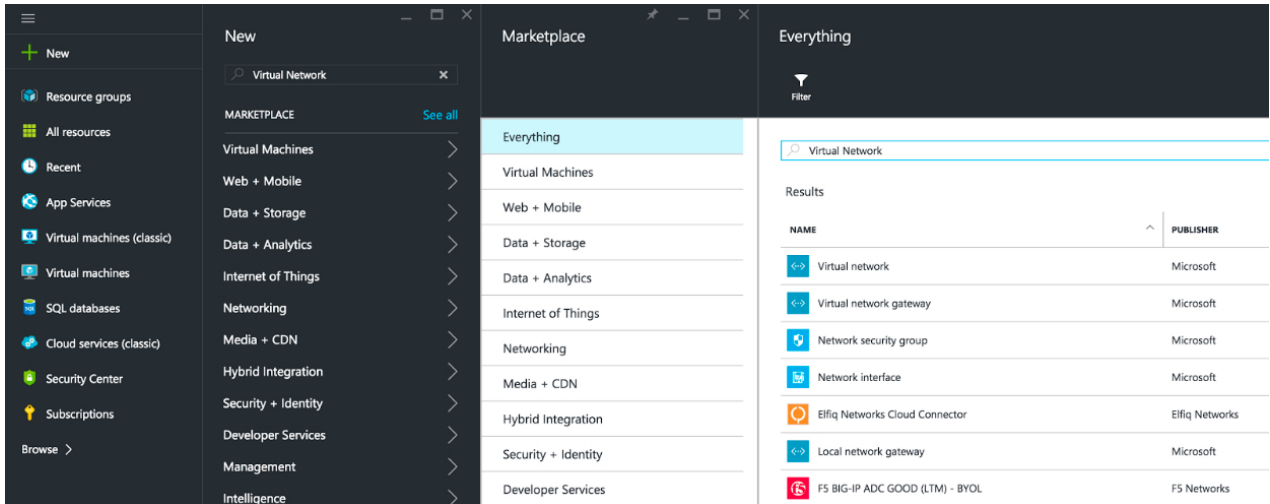
Setting Up a Virtual Network (VNet) and Subnet

Cloudera Director requires a virtual network and subnet to implement its networking environment. The networking environment must be set up for forward and reverse hostname resolution. We provide a basic example for setting up forward and reverse hostname resolution in [Setting Up Dynamic DNS on Azure](#) on page 53.

Read this [Azure document](#) for an overview of virtual networks on Azure.

To set up a new virtual network and its subnets, follow the steps below. Skip these steps if you are using an existing virtual network and subnet.

1. In the left pane, click **New**.
2. Type `Virtual Network` in the search box.
3. Click **Virtual Network** in the search result.
4. Click **Create**.
5. Type in a name for the virtual network and subnet
6. Type in a name for new resource group or select an existing resource group.
7. Click **Create**.



Setting Up Availability Sets for Master Nodes and Worker Nodes

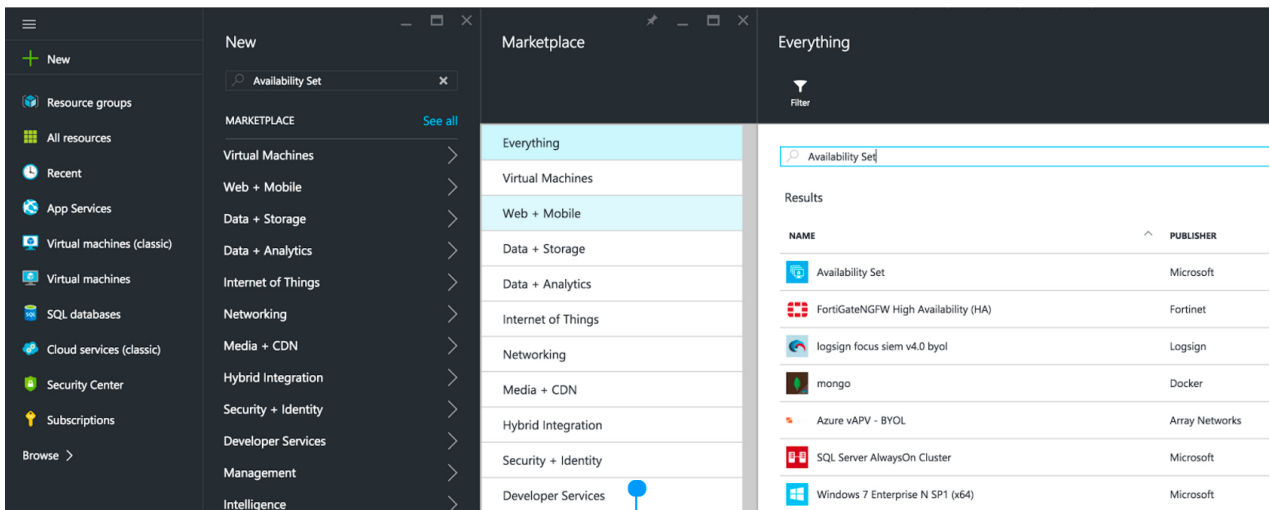
Azure uses availability sets as the tool to manage the availability of virtual machines. For best practices, in a CDH cluster, Cloudera recommends using one availability set for the master nodes and one availability set for the worker nodes. An availability set should not be shared by more than one CDH cluster.

Read this [Azure document](#) for an overview of availability sets on Azure.

To create an availability set:

1. In the left pane, click **New**.
2. Type `Availability Set` in the search box.
3. Click **Availability Set** in the search result.
4. Click **Create**.
5. Type in a name for the availability set.
6. Type in a name for new resource group or select an existing resource group.
7. Increase the fault domain and update domain to as large a size as possible.
8. Click **Create**.

After performing the above steps to create the availability set for master nodes, repeat them to create an availability set for worker nodes.



Setting Up Dynamic DNS on Azure

Overview

Running Hadoop (specifically CDH, in this case) requires forward and reverse DNS for internal IP addresses, something not currently supported in Microsoft Azure. This means you must use your own DNS server to run CDH on Azure. For more information on using your own DNS server on Azure, see [Name resolution using your own DNS server](#) in the Azure documentation. Below is a basic example for setting up a Dynamic DNS (DDNS) server to provide forward and reverse hostname resolution.



Important: If you are already using your own DNS server, ensure that it supports DNS reverse lookup and skip this section.

This section provides steps for:

- Setting up basic DDNS using BIND
- Required configuration and zone files
- Update scripts that will automatically update BIND when IP addresses are assigned or changed (for example, when stopping and starting hosts)

There are places where this document assumes certain configurations and architecture and those assumptions are noted.

Getting Started with Cloudera Director

The DNS Server and the Cloudera Director Host

Creating a DNS Server and Cloudera Director Host

This example shows setting up the DNS server and Cloudera Director to run on the same host.

Creating a Virtual Machine for the DNS Server

In Azure, select or create the resource group you will be using for your cluster. Select the + button to add a resource within that resource group. Search for the VM image CDH `cloudera-centos` and create it following the instructions in [Setting Up a Virtual Machine for Cloudera Director Server](#). Make sure port 53 is accessible on the VM intended to be the DNS server.

Selecting DNS Defaults

Pick an internal host FQDN suffix. This is the suffix for all internal hostname resolution within Cloudera clusters and is the same thing that is asked for when setting up clusters via Cloudera Director:

Host FQDN suffix * ?

This is entirely based on your environment. Examples include `cdh-cluster.internal`, `cluster.company-name.local`, and `internal.company-name.com`.



Note: We provide a set of scripts on the Cloudera GitHub site to automate the BIND install and setup process: <https://github.com/cloudera/director-scripts/tree/master/azure-dns-scripts>. The provided scripts are *not* intended for setting up BIND for production use.

Setting Up BIND on the Host

This section describes how to set up BIND on the host.

Information from Azure

The sample BIND files use this information. Modify the values in this example for your environment:

- Hostname: `director`
- Virtual Network Address Space: `10.3.0.0/16`
- Private IP: `10.3.0.4`

Installing BIND

Perform the following changes as root. Either run after `sudo -i` or start all commands with `sudo`.

```
# install bind
yum -y install bind bind-utils

# make the directories that bind will use
mkdir /etc/named/zones
# make the files that bind will use
touch /etc/named/named.conf.local
touch /etc/named/zones/db.internal
touch /etc/named/zones/db.reverse
```

Updating or Creating the Files

The contents of each of the four files and the changes needed are included in-place below. See the comments inline for changes you need to make. The following changes need to be performed as root. Either run after `sudo -i` or start all commands with `sudo`.

`/etc/named.conf`

```
//
// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS
// server as a caching only nameserver (as a localhost DNS resolver only).
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
```

```
//
acl trusted {
    // replace `10.3.0.0/16` with your subnet
    10.3.0.0/16;
};

options {
    // replace `10.3.0.4` with the internal IP of the BIND host
    listen-on port 53 { 127.0.0.1; 10.3.0.4; };
    listen-on-v6 port 53 { ::1; };
    directory "/var/named";
    dump-file "/var/named/data/cache_dump.db";
    statistics-file "/var/named/data/named_stats.txt";
    memstatistics-file "/var/named/data/named_mem_stats.txt";
    allow-query { localhost; trusted; };
    recursion yes;
    forwarders { 168.63.129.16; }; // used for all regions
    dnssec-enable yes;
    dnssec-validation yes;
    dnssec-lookaside auto;

    /* Path to ISC DLV key */
    bindkeys-file "/etc/named.iscdlv.key";

    managed-keys-directory "/var/named/dynamic";
};

logging {
    channel default_debug {
        file "data/named.run";
        severity dynamic;
    };
};

zone "." IN {
    type hint;
    file "named.ca";
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
include "/etc/named/named.conf.local";
```

`/etc/named/named.conf.local`

```
// replace the zone name (`cdh-cluster.internal`) with with the internal host FQDN suffix
// you wish to use for your cluster network (this is an option exposed in Director)
zone "cdh-cluster.internal" IN {
    type master;
    file "/etc/named/zones/db.internal";
    // replace with your subnet
    allow-update { 10.3.0.0/16; };
};

// replace the zone name (`0.3.10.in-addr.arpa`) with the network component of your
// subnet, reversed
// (example: with a subnet definition of 10.3.0.0/24, the reversed subnet component
// would be 0.3.10)
zone "0.3.10.in-addr.arpa" IN {
    type master;
    file "/etc/named/zones/db.reverse";
    // replace with your subnet
    allow-update { 10.3.0.0/16; };
};
```

`/etc/named/zones/db.internal`

```
$ORIGIN .
$TTL 600 ; 10 minutes
; replace `cdh-cluster.internal` with the zone name defined in
/etc/named/named.conf.local)
; replace `director.cdh-cluster.internal` with the internal fqdn of the primary name
server; note the trailing period (`.`)
; replace `hostmaster.cdh-cluster.internal` with the hostmaster email address, represented
with only periods (.), by convention this is `hostmaster.<your fqdn suffix>`; note the
trailing period (.)
cdh-cluster.internal IN SOA director.cdh-cluster.internal.
hostmaster.cdh-cluster.internal. (
    10 ; serial
    600 ; refresh (10 minutes)
    60 ; retry (1 minute)
    604800 ; expire (1 week)
    600 ; minimum (10 minutes)
)
; replace `director.cdh-cluster.internal` with the internal fqdn of the primary
name server; note the trailing period (.)
NS director.cdh-cluster.internal.

; replace `cdh-cluster.internal` with the zone name defined in
/etc/named/named.conf.local; note the trailing period (.)
$ORIGIN cdh-cluster.internal.
; replace `director` with the hostname of your DNS host, this should be the prefix of
the internal fqdn of the primary name server
; replace `10.5.0.4` with the internal IP of the primary name server
director A 10.5.0.4
```

`/etc/named/zones/db.reverse`

```
$ORIGIN .
$TTL 600 ; 10 minutes
; replace `0.5.10.in-addr.arpa` with the the network component of your subnet, reversed
(the zone name defined in /etc/named/named.conf.local)
; replace `director.cdh-cluster.internal` with the internal fqdn of the primary name
server; note the trailing period (.)
; replace `hostmaster.cdh-cluster.internal` with the hostmaster email address, represented
with only periods (.), by convention this is `hostmaster.<your fqdn suffix>`; note the
trailing period (.)
0.5.10.in-addr.arpa IN SOA director.cdh-cluster.internal.
hostmaster.cdh-cluster.internal. (
    10 ; serial
    600 ; refresh (10 minutes)
    60 ; retry (1 minute)
    604800 ; expire (1 week)
    600 ; minimum (10 minutes)
)
; replace `director.cdh-cluster.internal` with the internal fqdn of your primary
name server; note the trailing period (.)
NS director.cdh-cluster.internal.

; replace `0.5.10.in-addr.arpa` with the the network component of your subnet, reversed
(the zone name defined in /etc/named/named.conf.local)
$ORIGIN 0.5.10.in-addr.arpa.
; replace `4` with the host number of the private IP of your DNS host
; replace `director.cdh-cluster.internal` with the internal fqdn of your primary name
server
4 PTR director.cdh-cluster.internal.
```

Checking BIND Configuration

The syntax of BIND configuration files must be exact. Before starting the nameserver, check that the BIND configuration is valid.

```
# named-checkconf /etc/named.conf
```

Correct any errors (blank output means no errors).

Starting BIND

1. `chown /etc/named*` to `named:named` (named needs read/write privileges here:

```
# chown -R named:named /etc/named*
```

2. start bind:

```
# service named start
```

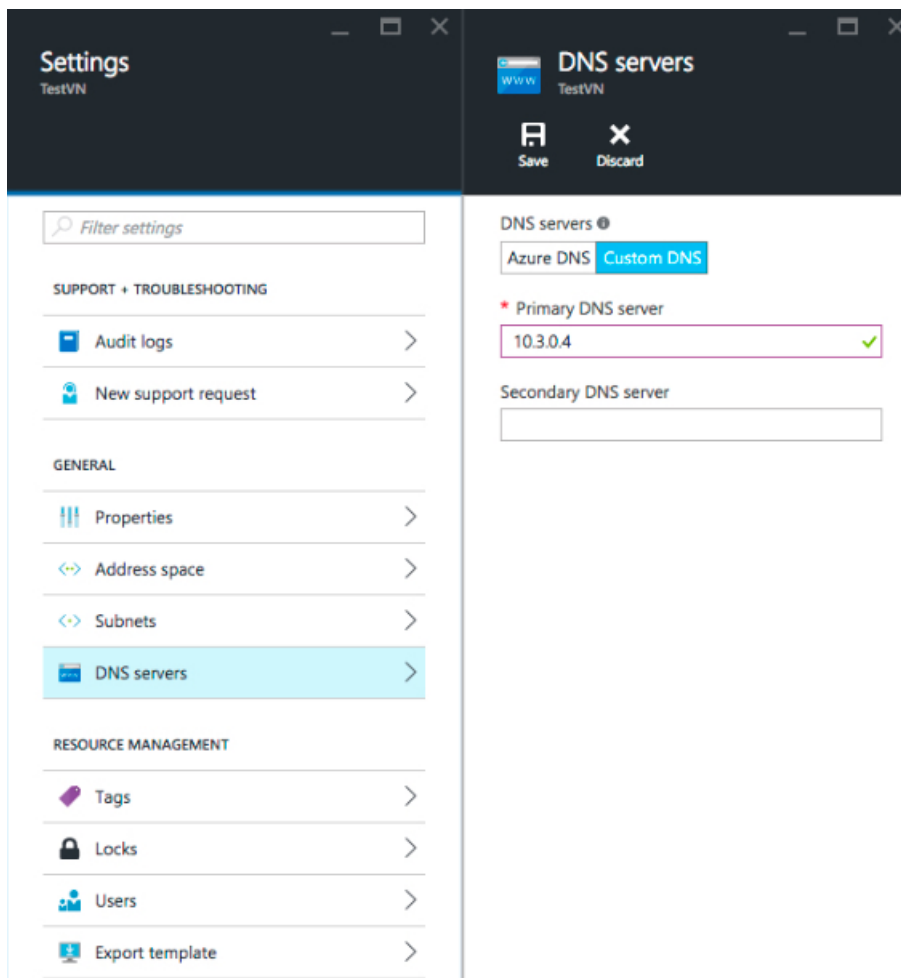
3. Set bind to start on startup:

```
# chkconfig named on
```

Swapping DNS from Azure to BIND

To change the DNS settings on Azure:

1. In the left pane, click on **Resource groups**.
2. Select the resource group your DNS server is in.
3. Click on the virtual network your cluster is using.
4. Click settings.
5. Click **DNS servers**.
6. Set **DNS servers** to **Custom DNS**.
7. Set **Primary DNS server** to the private IP address of your Cloudera Director host (10.3.0.4 in this example).



Wait for the DNS setting update to complete in the Azure portal, then restart the network service on the VM. VMs created after the DNS setting is updated in the Azure portal will automatically pick up the new DNS server address.

Restart the network service to pull down the nameserver changes entered in the Azure portal:

```
service network restart
```

If the change has propagated, the `nameserver` entry in `/etc/resolv.conf` will be what you entered in the Azure portal:

```
cat /etc/resolv.conf
```

If the change has not yet propagated, wait two minutes and restart the network service again (you may have to do this multiple times).

RHEL 6 and CentOS 6: Add `dhclient-exit-hooks`

This script will create a new `dhclient-exit-hooks` file in `/etc/dhcp/` and set the file to be executable. Run the script as root:

```
#!/bin/sh
# cat a here-doc representation of the hooks to the appropriate file
cat > /etc/dhcp/dhclient-exit-hooks <<"EOF"
#!/bin/bash
printf "\ndhclient-exit-hooks running...\n\treason:%s\n\tinterface:%s\n" "${reason:?}"
"${interface:?}"
# only execute on the primary nic
if [ "$interface" != "eth0" ]
then
    exit 0;
fi
# when we have a new IP, perform nsupdate
if [ "$reason" = BOUND ] || [ "$reason" = RENEW ] ||
[ "$reason" = REBIND ] || [ "$reason" = REBOOT ]
then
    printf "\tnew_ip_address:%s\n" "${new_ip_address:?}"
    host=$(hostname | cut -d'.' -f1)
    domain=$(hostname | cut -d'.' -f2- -s)
    domain=${domain:='cdh-cluster.internal'} # If no hostname is provided, use
cdh-cluster.internal
IFS='.' read -ra ipparts <<< "$new_ip_address"
ptrrec="${ipparts[3]}.${ipparts[2]}.${ipparts[1]}.${ipparts[0]}.in-addr.arpa"
nsupdatecmds=$(mktemp -t nsupdate.XXXXXXXXXX)
resolvconfupdate=$(mktemp -t resolvconfupdate.XXXXXXXXXX)
echo updating resolv.conf
grep -iv "search" /etc/resolv.conf > "$resolvconfupdate"
echo "search $domain" >> "$resolvconfupdate"
cat "$resolvconfupdate" > /etc/resolv.conf
echo "Attempting to register $host.$domain and $ptrrec"
{
    echo "update delete $host.$domain a"
    echo "update add $host.$domain 600 a $new_ip_address"
    echo "send"
    echo "update delete $ptrrec ptr"
    echo "update add $ptrrec 600 ptr $host.$domain"
    echo "send"
} > "$nsupdatecmds"
nsupdate "$nsupdatecmds"
fi
#done
exit 0;
EOF
chmod 755 /etc/dhcp/dhclient-exit-hooks
service network restart
```

RHEL 7 and CentOS 7: Add NetworkManager Dispatcher Scripts

This script will create an `/etc/NetworkManager/dispatcher.d/12-register-dns` file and set the file to be executable. Run the script as root:

```
#!/bin/sh
# RHEL 7.2 uses NetworkManager. Add a script to be automatically invoked when interface
# comes up.
cat > /etc/NetworkManager/dispatcher.d/12-register-dns <<"EOF"
#!/bin/bash
# NetworkManager Dispatch script
# Deployed by Cloudera Director Bootstrap
#
# Expected arguments:
#   $1 - interface
#   $2 - action
#
# See for info: http://linux.die.net/man/8/networkmanager

# Register A and PTR records when interface comes up
# only execute on the primary nic
if [ "$1" != "eth0" || "$2" != "up" ]
then
    exit 0;
fi

# when we have a new IP, perform nsupdate
new_ip_address="$DHCP4_IP_ADDRESS"

host=$(hostname -s)
domain=$(hostname | cut -d'.' -f2- -s)
domain=${domain:='cdh-cluster.internal'} # REPLACE-ME If no hostname is provided, use
cdh-cluster.internal
IFS='.' read -ra ipparts <<< "$new_ip_address"
ptrrec="$(printf %s "$new_ip_address." | tac -s.)in-addr.arpa"
nsupdatecmds=$(mktemp -t nsupdate.XXXXXXXXXX)
resolvconfupdate=$(mktemp -t resolvconfupdate.XXXXXXXXXX)
echo updating resolv.conf
grep -iv "search" /etc/resolv.conf > "$resolvconfupdate"
echo "search $domain" >> "$resolvconfupdate"
cat "$resolvconfupdate" > /etc/resolv.conf
echo "Attempting to register $host.$domain and $ptrrec"
{
    echo "update delete $host.$domain a"
    echo "update add $host.$domain 600 a $new_ip_address"
    echo "send"
    echo "update delete $ptrrec ptr"
    echo "update add $ptrrec 600 ptr $host.$domain"
    echo "send"
} > "$nsupdatecmds"
nsupdate "$nsupdatecmds"
exit 0;
EOF
chmod 755 /etc/NetworkManager/dispatcher.d/12-register-dns
service network restart
```

Checking DNS

Azure has hooks to automatically overwrite `/etc/resolv.conf` with Azure-specific values. However, depending on OS, the contents of `/etc/dhcp/dhclient-exit-hooks` or `/etc/NetworkManager/dispatcher.d/12-register-dns` are executed after Azure's hooks, and so can overwrite `/etc/resolv.conf` with custom values.

If you `cat /etc/resolv.conf` it should appear as follows:

```
; generated by /sbin/dhclient-script
nameserver 10.3.0.4
search cdh-cluster.internal
```

Getting Started with Cloudera Director

You should now be able to resolve internal FQDNs and do forward and reverse DNS queries without errors:

```
# hostname -f
director.cdh-cluster.internal

# hostname -i
10.3.0.4

# host `hostname -i`
4.0.3.10.in-addr.arpa domain name pointer director.cdh-cluster.internal

# host `hostname -f`
director.cdh-cluster.internal has address 10.3.0.4
```

Note that the values 10.3.0.4, 4.0.3.10, and cdh-cluster.internal are specific to this example and will be different for you.

Errors like the following indicate that there is a problem with the DNS configuration:

```
# hostname -f
hostname: Unknown host

# hostname -i
hostname: Unknown host

# host `hostname -i`
Host 4.0.3.10.in-addr.arpa. not found: 3(NXDOMAIN)
```

Setting Up MySQL or PostgreSQL

A database server can be installed on the same host as Cloudera Director and DNS, or you can add your database server to a different host in the same virtual network as Cloudera Director and the cluster. The supported databases are MySQL and PostgreSQL.

A dedicated database server is required for production clusters. The following steps are optional for non-production proof-of-concept clusters.

Database Server Requirements

A database server can be installed on the same host as Cloudera Director and DNS, or you can add your database server to a different host in the same virtual network as Cloudera Director and the cluster. The supported databases are MySQL and PostgreSQL.

- The database server must be JDBC accessible both locally and remotely
- The credentials provided to Cloudera Director must have superuser/administrator privileges
- Increase the connection count according to Cloudera's documentation on [MySQL Database](#) or [PostgreSQL Database](#)
- Ensure sufficient CPU, memory, IOs, and bandwidth for the database server, especially if the database server is shared between multiple clusters

Example

Using MySQL as an example, follow the instructions in [MySQL Database](#). Be sure to use the instructions for your specific version of MySQL, and keep in mind these extra requirements:

- Your MySQL server must be in the same virtual network as the rest of the cluster.
- Reference your MySQL server host by private IP address or an internal fully-qualified domain name that resolves to a private IP address.
- You must run the same `dhclient` script on this host as well.

Setting Up a Virtual Machine for Cloudera Director Server

Cloudera Director server is used to provision CDH clusters. See the Azure document [Create a Linux VM on Azure using the Portal](#) for an overview on creating a Linux VM on Azure. We recommend using the CentOS image published by Cloudera on Marketplace.

Provide the following values during creation:

- Instance size should be D3 or larger.
- Typically, install Cloudera Director in the same virtual network and subnet of the cluster.
- Typically, specify the same network security group.
- Typically, set the same availability as you will set on the master nodes.
- A Public IP address is optional, depending on the access pattern you use.

Installing Cloudera Director Server and Client on Azure

To install Cloudera Director, perform the following tasks. You must be either running as root or using sudo to perform these tasks.

RHEL 7 and CentOS 7

1. SSH to the Azure instance you created for Cloudera Director.
2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For download and installation information, see [Java SE Downloads](#).

```
sudo yum localinstall jdk-version-linux-x64.rpm
```

3. Add the Cloudera Director repository to the package manager:

```
cd /etc/yum.repos.d/
sudo wget
"http://archive.cloudera.com/director/redhat/7/x86_64/director/cloudera-director.repo"
```

4. Install Cloudera Director server and client by running the following command:

```
sudo yum install cloudera-director-server cloudera-director-client
```

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

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

6. Disable and stop the firewall with the following commands:

```
sudo systemctl disable firewalld
sudo systemctl stop firewalld
```

RHEL 6 and CentOS 6

1. SSH to the Azure instance you created for Cloudera Director.
2. Install a supported version of the Oracle Java Development Kit (JDK) on the Cloudera Director host. Currently, Cloudera Director supports JDK versions 7 and 8. For download and installation information, see [Java SE Downloads](#).

```
sudo yum localinstall jdk-version-linux-x64.rpm
```

3. Add the Cloudera Director repository to the package manager:

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

4. Install Cloudera Director server and client by running the following command:

```
sudo yum install cloudera-director-server cloudera-director-client
```

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

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

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

```
sudo service iptables save
sudo chkconfig iptables off
sudo service iptables stop
```

Sample Configurations

Three sample configuration files are available on the [Cloudera GitHub site](#). You can modify these sample director configuration files to create clusters using the Cloudera Director CLI.

- [azure.simple.conf](#): This is a simple Cloudera Director configuration that creates a Cloudera Manager node and a four-node cluster (one master and three workers).
- [azure.reference.conf](#): This is a reference Cloudera Director configuration that creates an eight-node cluster (three masters and five workers) with high availability (HA) enabled.
- [azure.kerberos.conf](#): This is the same Cloudera Director configuration as the [azure.reference.conf](#) configuration, but with Kerberos enabled.

Configuring a SOCKS Proxy for Microsoft Azure

For security purposes, Cloudera recommends that you connect to your cluster using a [SOCKS proxy](#). A SOCKS proxy changes your browser to do lookups directly from your Microsoft Azure network and allows you to connect to services using private IP addresses and internal FQDNs.

This approach will do the following:

- Set up a single SSH tunnel to one of the hosts on the network (the Cloudera Director host in this example), and create a SOCKS proxy on that host.
- Change the browser configuration to do all lookups via that SOCKS proxy host.

Network Prerequisites

The following are prerequisites for connecting to your cluster using a SOCKS proxy:

- The host that you proxy to must be reachable from the public internet (or the network that you're connecting from).
- The host that you proxy to must be able to reach the Cloudera Director server via private IP (proxying directly to the Cloudera Director server works as well).

Start the SOCKS Proxy

To start a SOCKS5 proxy over SSH run the following command:

```
ssh -i your-key-file.pem -CND 1080
the_username_you_specified@instance_running_director_server
```

The parameters are as follows:

- `-i your-key-file.pem` specifies the path to the private key needed to ssh to the Cloudera Director server
- `C` sets up compression
- `N` suppresses any command execution once established
- `D` sets up the SOCKS proxy on a port
- `1080` is the port to set the SOCKS proxy locally

Configure Your Browser to Use the Proxy

Next, configure your browser settings to use the socks proxy.

On Google Chrome

By default, Google Chrome uses system-wide proxy settings on a per-profile basis. To get around that we will launch Chrome via the command line and specify the following:

- The SOCKS proxy port to use (this must be the same value used above)
- The profile to use (this example will create a new profile)

This will create a new profile and launch a new instance of Chrome that won't interfere with your current running instance of Chrome.

Linux

```
/usr/bin/google-chrome \
--user-data-dir="$HOME/chrome-with-proxy" \
--proxy-server="socks5://localhost:1080"
```

Mac OS X

```
"/Applications/Google Chrome.app/Contents/MacOS/Google Chrome" \
--user-data-dir="$HOME/chrome-with-proxy" \
--proxy-server="socks5://localhost:1080"
```

Microsoft Windows

```
"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" ^
--user-data-dir="%USERPROFILE%\chrome-with-proxy" ^
--proxy-server="socks5://localhost:1080"
```

Now in this Chrome session you can connect to any Cloudera Director accessible host using the private IP address or internal FQDN. For example, if you proxy to the Cloudera Director server, you can connect to Cloudera Director as if it were local by putting `localhost:7189` in Chrome's URL bar.

Allowing Access to VM Images

The Cloudera Director Azure plugin deploys Azure VM images programmatically. In order to allow programmatic deployment of VM images on Azure, the user must accept a term of usage and grant their Azure subscription permission to deploy the VM images. Detailed steps for allowing programmatic deployment of Azure VM images are available at [Working with Marketplace Images on Azure Resource Manager](#).

By default the Cloudera Director Azure plugin uses the Cloudera-certified CentOS 6 image. Follow the instructions from the above link to allow programmatic deployment of the `cloudera-centos-6` image.

Microsoft Azure > New > Marketplace > Everything > Cloudera CentOS 6.7 > Configure Programmatic Deployment

Cloudera CentOS 6.7

Bring Your Own License enabled.

Cloudera Enterprise helps you become information-driven by leveraging the best of the open source community with the enterprise capabilities you need to succeed with Apache Hadoop in your organization. Designed specifically for mission-critical environments, Cloudera Enterprise includes CDH, the world's most popular open source Hadoop-based platform, as well as advanced system management and data management tools plus dedicated support and community advocacy from our world-class team of Hadoop developers and experts. Cloudera is your partner on the path to big data. Cloudera Enterprise, with Apache Hadoop at the core, is: Unified - one integrated system, bringing diverse users and application workloads to one pool of data on common infrastructure; no data movement required Secure - perimeter security, authentication, granular authorization, and data protection Governed - enterprise-grade data auditing, data lineage, and data discovery Managed - native high-availability, fault-tolerance and self-healing storage, automated backup and disaster recovery, and advanced system and data management Open - Apache-licensed open source to ensure your data and applications remain yours, and an open platform to connect with all of your existing investments in technology and skill.

This is the OS image that enables installation of Cloudera. This does not install Cloudera by itself. Cloudera distribution bundles the innovative work of a global open-source community, including critical bug fixes and important new features from the public development repository, and applies it to a stable version of the source code. In short, Cloudera integrates the most popular projects related to Hadoop into a single package that is rigorously tested to ensure reliability during production.

PUBLISHER: Cloudera

Select a deployment model Resource Manager

Create

Want to deploy programmatically? [Get started →](#)

Configure Programmatic Deployment

Use API calls, ARM templates, or the PowerShell console to automatically deploy without using the Azure portal. You'll only need to do this once—the settings you choose will be used each time you deploy.

Offer details

Cloudera CentOS 6.7 by Cloudera
[Terms of use](#) | [privacy policy](#)

Pricing does not include **Azure infrastructure costs** (e.g., virtual machine compute time or storage) and is based on the pricing tier you select at the time of deployment. The pricing above applies only to Azure subscriptions purchased from Microsoft. For Azure subscriptions purchased from a reseller, contact your reseller for pricing. Neither subscription credits nor monetary commitment funds may be used to purchase non-Microsoft offerings. These purchases are billed separately. If any Microsoft products are included in the above offering(s) (e.g., Windows Server or SQL Server), such products are licensed by Microsoft and not by any third party.

Terms of use

By enabling programmatic purchases for the subscriptions selected below, I (a) agree to the legal terms and privacy statement(s) associated with each offering above, (b) for Azure subscriptions purchased from Microsoft, authorize Microsoft to charge or bill my current payment method for the fees associated with my use of the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s), and (c) agree that Microsoft may share my contact information, and transaction details associated with my purchase of the above offering(s), with any third-party vendors, if listed above. Microsoft does not provide rights for third-party products or services. See the [Azure Marketplace Terms](#) for additional terms.

Choose the subscriptions

Select the Azure subscriptions for which you would like to enable programmatic deployments of the above offering(s)

SUBSCRIPTION NAME	SUBSCRIPTION ID	STATUS
Pay-As-You-Go 1	00000000-0000-0000-0000-000000000000	<input type="checkbox"/> Enable <input type="checkbox"/> Disable
Pay-As-You-Go 2	00000000-0000-0000-0000-000000000000	<input type="checkbox"/> Enable <input type="checkbox"/> Disable

Save **Discard**

Creating a Cluster

Before You Deploy Cloudera Manager and CDH



Important: Make sure at least one VM has been manually deployed from the Azure portal into the Azure subscription you intend to use for your cluster before using Cloudera Director to deploy clusters.

This page contains steps for setting up Cloudera Manager and a CDH cluster in Microsoft Azure using the Cloudera Director web UI. This initial section lists the requirements that must be met before beginning the deployment procedure:

- Create an AD application and a service principal for the AD application.
 - The AD application must have the **contributor** role or similar role so that it has permission to create and delete resources in the subscription.
- A Virtual Network and Network Security Group must be created or readily available for the cluster to use.
- The Virtual Network must be configured to use a customer-provided DNS service that supports reverse lookup.
 - If using the provided DNS service setup guide, the VM that provides the DNS service must be created and running.
- Resource Group(s) created to house cluster VMs are required.
- An availability set created in corresponding Resource Groups to house cluster VMs.
- Cloudera Director server VM created.
- Cloudera Director server installed and running.
- Cloudera Director server has access to the VNet.
- Database server must be created or readily available.
- The database server must be reachable from the VNet to be used by cluster nodes.

Details of setting up individual items above is covered in earlier sections.

Deploying Cloudera Manager and CDH on Microsoft Azure

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

To create an environment:

1. Open a web browser and go to the private IP address of the instance you created running Cloudera Director server. 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:
 - a. Enter a name in the **Environment Name** field.
 - b. In the **Cloud provider** field, select **Azure Cloud Platform**.
 - c. In the **Management URL** field, enter the Azure management URL provided by Microsoft. You don't need to change the default value unless you are in an Azure region that uses a different URL.
 - d. In the **Subscription ID** field, enter the Azure subscription ID.
 - e. In the **AAD URL** field, enter the Azure Active Directory (AAD) URL provided by Microsoft. You don't need to change the default value unless you are in an Azure region that uses a different URL.
 - f. In the **Tenant ID** field, enter the AAD tenant ID of your ADD tenant. See Obtain [Obtaining Credentials for Cloudera Director](#) for details on obtaining the AAD tenant ID.
 - g. In the **Client ID** field, enter the client ID of the Azure service principal you created earlier. See [Obtaining Credentials for Cloudera Director](#) for details on obtaining the client ID.
 - h. In the **Client Secret** field, enter the client secret of the Azure service principal you created earlier. See [Obtaining Credentials for Cloudera Director](#) for details on obtaining the client secret.
 - i. In the **Advanced Options** area, select the Azure region where your cluster is located from the drop down list.
 - j. In the **Advanced Options** section, enter the Azure region where your Cloudera Director instance is located.
 - k. In the **SSH Credentials** section:
 - a. Enter a username in the **Username** field. Azure will create the user specified here.
 - b. Create an SSH key with the following command:

```
ssh-keygen -f ~/.ssh/my_azure_vm_keyname -t rsa
```

- c. Copy the SSH private key into the **Private key** field. Cloudera Director uses the SSH key pairs to create and access VMs in Azure.

Add Environment

GENERAL INFORMATION

Environment name * ?

Cloud provider Microsoft Azure Cloud Platform ▼ ?

Management URL * ?

Subscription ID * ?

AAD URL * ?

Tenant ID * ?

Client ID * ?

Client Secret * ?

AZURE

▼ Advanced Options

Region ▼ ?

SSH CREDENTIALS

Username * ?

Private key * File Upload Direct Input ?

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

6. In the **Add Cloudera Manager** screen:
 - a. Enter a name for this deployment of Cloudera Manager in the **Cloudera Manager name** field.
 - b. In the **Instance Template** field, select **Create New Instance Template**.
 - c. The **Instance Template** model screen displays.
7. In the **Instance Template** model screen, do the following:
 - a. In the **Instance Template** name field, enter a name for the template.
 - b. In the **Instance Type** field, select **STANDARD_DS13** or **STANDARD_DS14**.
 - c. In the **Image Alias** field, select `cloudera-centos-6-latest`.
 - d. In the **Tags** field, add one or more tags to associate with the instance.
 - e. In the **Compute Resource Group** field, enter the name of the resource group you created earlier to house the VM.
 - f. In the **Virtual Network Resource Group** field, enter the name where the virtual network resource resides.
 - g. In the **Virtual Network** field, enter the name of the virtual network.
 - h. In the **Subnet Name** field, enter the name of the subnet you wish to use.
 - i. In the **Host FQDN suffix** field, enter the name of the host FQDN suffix you would like your cluster host to use. This is the DNS domain of your cluster hosts.
 - j. In the **Network Security Group Resource Group** field, enter the name of the resource group where the network security group resource resides.
 - k. In the **Network Security Group** field, enter the name of the network security group.
 - l. Select **Yes** in the **Public IP** field if you want to assign a public IP address to the VM. The default value is **No**.
 - m. In the **Availability Set** field, enter the name of the availability set you created in earlier steps.
 - n. In the **Instance name prefix** field under **Advanced Options**, enter the desired instance name prefix.
 - o. In the **Data Disk Count** field in **Advanced Options**, enter the desired number of data disks to attach for the VM.
 - p. In the **Bootstrap script** field in **Advanced Options**, paste in or upload the desired custom bootstrap script.



Important: If you created a DNS service following the DNS service setup guide, use this [bootstrap script](#) to ensure that the DNS record is updated correctly.

Create New Instance Template
✕

Instance Template name *

?

VirtualMachine Size *

?

Image Alias *

?

Tags

?

Compute Resource Group *

?

Virtual Network Resource Group *

?

Virtual Network *

?

Subnet Name *

?

Host FQDN suffix *

?

Network Security Group Resource Group *

?

Cancel
Save Changes

Public IP *

?

Availability Set *

?

▼ Advanced Options

Instance name prefix

?

Data Disk Count

?

SSH username

?

Bootstrap script
 File Upload
 Direct Input
 ?

8. In the **Desired License Type** field, select one of the following license types:

- Cloudera Enterprise: includes the core CDH services (HDFS, Hive, Hue, MapReduce, Oozie, Sqoop, YARN, and ZooKeeper) and, depending on the license edition, one or more additional services (Accumulo, HBase, Impala, Navigator, Solr, Spark). For more information on Cloudera Enterprise licenses, see [Managing Licenses](#) in the Cloudera Manager documentation.

- Cloudera Enterprise Trial: a 60-day trial license that includes all CDH services.
- Cloudera Express: no license required.

Licensing

Desired License Type * ?

Please provide a Cloudera Manager license key.

License Key * File Upload Direct Input ?

Billing ID ?

To enable usage-based billing, you must have a Cloudera Enterprise license and a billing ID provided by Cloudera. Perform these steps in the **Add Cloudera Manager** screen:

1. In the **Desired License Type** field, select **Cloudera Enterprise**.
 2. In the **License Key** field, either select a Cloudera Enterprise license file to upload or select **Direct Input** and input the license file text directly into the text area.
 3. To enable usage-based billing, enter the billing ID provided to you by Cloudera in the **Billing ID** field.
9. By default, the version of Cloudera Manager installed depends on the version of Cloudera Director you are using:
- If you are using Cloudera Director 2.0, the latest released version of Cloudera Manager 5.5 is installed by default.
 - If you are using Cloudera Director 2.1, the latest released version of Cloudera Manager 5.7 is installed by default.

To install a version of Cloudera Manager higher or lower than the default version, perform the following steps:

- a. In the **Configurations** section, check **Override default Cloudera Manager repository**.
- b. In the **Repository URL** field, enter the repository URL for the version of Cloudera Manager to install. Repository URLs for versions of Cloudera Manager 5 have the form <http://archive.cloudera.com/cm5/> followed by the operating system, operating system major version, processor architecture, cm (for Cloudera Manager), and the Cloudera Manager major, minor, and (if applicable) maintenance release number. For example, for Cloudera Manager 5.5.4, the repository URL is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.5.4/.



Note: The Cloudera Manager minor version must be the same as or higher than the CDH minor version. For example, Cloudera Manager 5.5 cannot be used to launch or manage a CDH 5.7 cluster, but Cloudera Manager 5.7 can be used with a CDH 5.7 (or lower) cluster.

- c. In the **Repository Key URL** field, enter the URL for the repository key. Repository key URLs have the same form as repository URLs except they end with the name of the key file instead of the Cloudera Manager version. For example, the repository key URL for any version of Cloudera Manager 5 on any supported version of Red Hat 7 is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera.
- 10 In the **Add Cloudera Manager** screen, click **Continue**.
- 11 At the **Confirmation** prompt, click **OK** to begin adding a cluster.
- 12 On the **Add Cluster** screen:
- a. Enter a name for the cluster in the **Cluster** name field.
 - b. Enter the version of CDH to deploy in the **Version** field or leave the default value. By default, the version of CDH installed depends on the version of Cloudera Director you are using:
 - If you are using Cloudera Director 2.0, the latest released version of CDH 5.5 is installed by default.

Getting Started with Cloudera Director

- If you are using Cloudera Director 2.1, the latest released version of CDH 5.7 is installed by default.

To install a version of CDH higher or lower than the default version, perform the following steps:

- a. Enter the desired CDH version in the **Version** field of the **Products** section. For example, for CDH 5.4.8 enter 5 . 4 . 8.
- b. Scroll down to **Configurations (optional)** and expand the section.
- c. Click **Override default parcel repositories**.
- d. Enter the repository parcel URL for the version of CDH you want to install. Parcel URLs for versions of CDH 5 have the form <http://archive.cloudera.com/cdh5/parcels/>, followed by the major, minor, and (if applicable) dot release number. For example, the URL for CDH 5.4.8 is <http://archive.cloudera.com/cdh5/parcels/5.4.8>.



Note: The CDH minor version must not be higher than the Cloudera Manager minor version. For example, CDH 5.7 will not work with Cloudera Manager 5.5, but CDH 5.7 (or lower) will work with Cloudera Manager 5.7.

- c. In the **Services** section, select the services you want to install.
- d. In the **Instance groups** area, create a new template for the groups or for each group and the number of instances you want.

Instance groups				
Name	Roles	Instance Template	Instance Count	
masters	Edit Roles	TEST-TEMPLATE Edit	1	Delete Group
workers	Edit Roles	TEST-TEMPLATE Edit	5	Delete Group
gateway	Edit Roles	TEST-TEMPLATE Edit	1	Delete Group
Add Group				

13 Click **Continue**.

14 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

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

Cleaning Up Your Azure Deployment

When you are done testing or using Cloudera Director, terminate your instances to stop incurring charges to your Azure account.

1. In Cloudera Director, terminate each instance in your clusters.
 - a. Click an environment name.
 - b. In the **Actions** column, select **Terminate Cluster**.
 - c. Repeat for each environment you configured.
2. If you want to save anything in Cloudera Director (the configuration file or database, for example), back it up.
3. In the Azure web UI, terminate the Cloudera Director instance and any other instance Cloudera Director was unable to terminate.

4. If applicable, terminate any external database you configured Cloudera Director to use.

Adding New VM Images

This section describes how to add new VM images. By default the Cloudera Director Azure plugin supports the following Cloudera-provided Azure VM image:

- `cloudera-centos-6-latest`

You can add additional images by modifying the `images.conf` configuration file in the plugin configuration directory. By default, the directory is at `/var/lib/cloudera-director-plugins/azure-provider-1.0.0/etc`. The configuration file looks as follows:

```
#
# Alias of configurable Azure VM Images
#

# This is the Centos 6 image published by Cloudera
cloudera-centos-6-latest {
  publisher: cloudera
  offer: cloudera-centos-6
  sku: CLOUDERA-CENTOS-6
  version: latest
}

# A new image must be specified as the following:
# new-image-name {
#   publisher: some_publisher
#   offer: offer_name
#   sku: sku_name
#   version: version_name
# }
```

Each Azure VM image is uniquely identified by four pieces of information:

1. publisher
2. offer
3. sku
4. version

To add a new image, create a new configuration block and provide the four required pieces of information. For more information on specifying images, see [Working with Marketplace Images on Azure Resource Manager](#) in the Microsoft Azure documentation. After updating the configuration file, restart the Cloudera Director server for the change to take effect.



Note: The Azure plugin is shipped with a default configuration file, `images.conf`, which is embedded in the plugin jar. If you specify a new configuration file, it will replace the one shipped with the plugin. This means if you want to use the default Cloudera VM images along with other third-party VM images, you must ensure that the Cloudera image information is retained in the new configuration file.

Important Notes

Azure Resources Managed by Cloudera Director

The Azure plugin for Cloudera Director creates the following resource:

- A storage account for each VM. Data drives created by Cloudera Director on Azure have a fixed size of 1 TB.
- A NIC for each VM.
- A public IP address for each VM, if public IP addresses are enabled.

Getting Started with Cloudera Director

Deploying Production Clusters

While the Cloudera Director web UI can be used for proof-of-concept deployments on Azure, you must use the published sample configuration files for production deployments (see [Useful Links](#) below). You can modify the sample configuration file to fit your specific deployment environment, remove services you don't need, and customize the sample bootstrap script. Configurations related to logging and data storage for individual services must not be changed. Deploying a cluster using the Cloudera Director command line interface and configuration file based on the examples ensures a repeatable deployment with the proper settings for Azure.

Refer to the [Cloudera Reference Architecture for Microsoft Azure Deployments](#) document for more details.

Deletion Behavior

The deletion behavior is as follows:

- The storage account created by the plugin is used for the VM OS drive and cluster data drive. If you have manually attached a drive from a different storage account not created by the plugin, it will not be deleted.
- The NIC created by the plugin is attached to the VM. We assume that only one NIC is used per VM. Do not manually attach NICs to the VM created by the plugin.
- Deleting the NIC also deletes the public IP (PIP) attached to the NIC. This includes PIPs created by Cloudera Director as well as PIPs attached manually.



Important: Because of the deletion behavior described above, do not reuse any resources created by the Azure plugin for any other purpose.

Useful Links

- [Cloudera Enterprise Reference Architecture for Azure Deployments.](#)
- [Configuration files for running Cloudera Director on Microsoft Azure:](#)
 - [azure.simple.conf](#): This is a simple Cloudera Director configuration that creates a Cloudera Manager node and a four-node cluster (one master and three workers).
 - [azure.reference.conf](#): This is a reference Cloudera Director configuration that creates an eight-node cluster (three masters and five workers) with high availability (HA) enabled.
 - [azure.kerberos.conf](#): This is the same Cloudera Director configuration as the [azure.reference.conf](#) configuration, but with Kerberos enabled.

Usage-Based Billing

Cloudera Director 2.1 and higher includes an automated metering service that enables usage-based billing, so that you only pay for the services you use. This section describes how usage-based billing works in Cloudera Director.

Prerequisites

The following are required for usage-based billing:

- Cloudera Director 2.1 or higher
- A billing ID provided by Cloudera. Your billing ID ensures that the Cloudera Manager instance and the clusters it manages are associated with your customer account, so that metering of your cluster usage is accurate.
- A Cloudera Enterprise license. When you provide a Cloudera Enterprise license and a billing ID during deployment of Cloudera Manager, usage-based billing is enabled for all clusters created with that Cloudera Manager instance. If you do not add a billing ID, usage-based billing is not enabled, and you are charged for your clusters under normal node-based billing.
- An account on a cloud service supported by Cloudera Director to deploy Cloudera Manager and CDH.
- Outbound HTTPS connectivity from Cloudera Director to Cloudera's metering service at <https://metering.cloudera.com> and the endpoints within AWS where usage information is collected. If outbound internet connectivity is restricted by your organization's security policies, then HTTPS connectivity can be narrowed to the [AWS IP address ranges](#).
- At least 2 GB of free disk space should be available on the Cloudera Director server to store usage information until it can be transmitted to the metering service.

How Usage-Based Billing Works

When usage-based billing is enabled, Cloudera Director collects cluster usage information at regular intervals in the form of usage bundles. The usage bundles are sent to a metering service that aggregates the information and determines the total bill.

The price for usage-based billing is determined by three factors:

- The Cloudera hourly rate, which is determined by two factors:
 - Instance type
 - CDH services enabled on the cluster
- Number of instances
- Number of hours

Hours billed are based on the time the instance or service starts, not on the time of day. Portions of an hour are rounded up to the next full hour. For example, an instance that runs from 1:40 pm. to 2:20 p.m. is charged for one hour.

Charging for instances in a cluster begins when bootstrapping is complete and the appropriate components have been installed and started on that cluster. The applicable rate is determined by the components that are deployed on the cluster for a given hour, so the price can change when a component is added or removed that would affect the rate.

There is no charge for instances in a cluster where none of the services are running, and billing stops for all instances in the cluster if the cluster is stopped or terminated. Billing and collection of usage information also stops if Cloudera Director is stopped. Billing resumes when Cloudera Director is started, but the billing hour for all billable clusters is reset from when Cloudera Director restarts.

The price charged for a running cluster depends partly on the CDH services it contains. The following table shows the five types of clusters defined for billing purposes, from least to most expensive.

Basic	Data Engineering	Operational DB	Analytic Database	Data Hub
"Core Hadoop"	"Core Hadoop" + Spark, Search	"Core Hadoop" + HBase, Spark, Search	"Core Hadoop" + Impala	All Capabilities

Usage-based billing only applies to your use of Cloudera Director, Cloudera Manager, and CDH services in the cloud. You are billed directly by your cloud provider for all cloud provider services, such as the virtual instances and databases used by your clusters.

Contact [Cloudera](#) for additional details about pricing with usage-based billing.

Deploying Cloudera Manager and CDH with Usage-Based Billing

When you create an instance of Cloudera Manager with a Cloudera Enterprise license and a billing ID, usage-based billing is enabled for all clusters you launch through that Cloudera Manager instance.

You can deploy Cloudera Manager and create clusters with usage-based billing either through the Cloudera Director server web UI or with the Cloudera Director client and the `bootstrap-remote` command, as described in this section.

Enabling Usage-Based Billing with the Cloudera Director Server web UI

The procedure for deploying Cloudera Manager and CDH through the Cloudera Director web UI is described in [Deploying Cloudera Manager and CDH on AWS](#) on page 32. To enable usage-based billing, follow the procedure as described there, but be sure to provide a Cloudera Enterprise license and a billing ID as described in the steps for the **Add Cloudera Manager** screen.

If you choose **Cloudera Enterprise**, the **License Key** and **Billing ID** fields are displayed. The **Billing ID** field is optional. Enter a valid license key, but do not enter a billing ID if you want your clusters to include Cloudera Enterprise features but without usage-based billing.



Note: If you deploy Cloudera Manager with a Cloudera Enterprise license but without a billing ID, you can add a billing ID later and launch clusters with usage-based billing. But you cannot add a Cloudera Enterprise license to an instance of Cloudera Manager that was created with a Cloudera Enterprise Trial or Cloudera Express license. If your Cloudera Manager instance does not have a Cloudera Enterprise license, you must deploy another Cloudera Manager instance *with* a Cloudera Enterprise license in order to use usage-based billing.

Licensing

Desired License Type * ?

Please provide a Cloudera Manager license key.

License Key * File Upload Direct Input ?

Billing ID ?

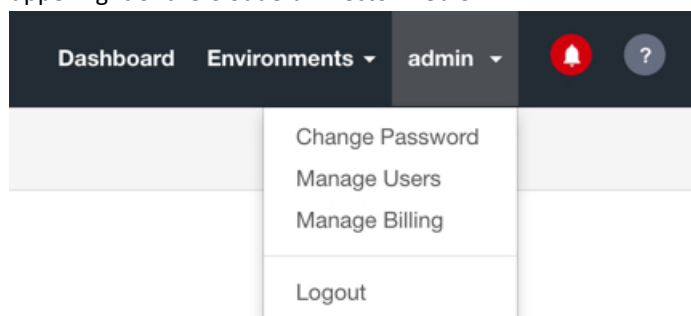
Enabling Usage-Based Billing with bootstrap-remote

The procedure for deploying Cloudera Manager and CDH through the Cloudera Director client using the `bootstrap-remote` command is described in [Submitting a Cluster Configuration File](#) on page 127.

There is a [sample Cloudera Director CLI configuration file](#) for remote bootstrapping a cluster on AWS with usage-based billing enabled. This configuration file will create a basic cluster with a Cloudera Enterprise license and billing ID. Edit the file to provide your license and billing ID, your credentials for your cloud provider, and configurations for your desired cluster services.

Managing Billing IDs with an Existing Deployment

To manage billing IDs for an existing deployment of Cloudera Manager, click **Manage Billing** on the admin menu in the upper right of the Cloudera Director web UI.



The Manage Billing page displays information about Cloudera Manager instances and environments managed by Cloudera Director.

If a Cloudera Manager instance has a Cloudera Enterprise license and a billing ID, the billing ID is displayed on this page in redacted form, as shown here for the Cloudera Manager instance CM01:

cloudera DIRECTOR					
Manage Billing					
Cloudera Manager	Environment	Status	License Type	Billing ID ?	Actions
CM01	NightlySandbox1 Environment	Ready	Cloudera Enterprise	*****k5epAAA	Update Billing ID ▼
NightlySandbox1 Deployment	NightlySandbox1 Environment	Ready	Cloudera Enterprise Trial	N/A	
NightlySandbox2 Deployment	NightlySandbox2 Environment	Ready	Cloudera Enterprise Trial	N/A	
NightlySandbox3 Deployment	NightlySandbox3 Environment	Ready	Cloudera Enterprise Trial	N/A	
CM02	NightlySandbox1 Environment	Ready	Cloudera Enterprise	Not Assigned	Assign Billing ID

If a Cloudera Manager deployment has a Cloudera Enterprise license but does not have a billing ID, as shown above for the deployment CM02, the value of the **Billing ID** for that instance is **Not Assigned** and usage-based billing is not enabled. You can add a billing ID for that Cloudera Manager deployment to enable usage-based billing. To add a billing ID to an existing Cloudera Manager deployment:

1. On the Manage Billing page, click **Assign Billing ID** to open the **Update Billing ID** dialog.

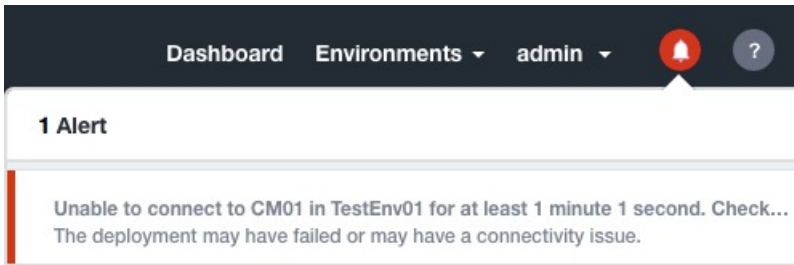
2. Enter a valid billing ID.
3. Click **Update**.

To replace a billing ID with a different one:

1. Click **Update Billing ID**.
2. In the **Update Billing ID** dialog, enter the new billing ID.
3. Click **Update**.

Troubleshooting Network Connectivity for Usage-Based Billing

If Cloudera Director is unable to connect to or upload usage information to the metering service, or is unable to connect to Cloudera Manager to obtain the usage information, an alert appears under the bell icon at the upper right of the top banner in the Cloudera Director web UI, and the bell icon turns red. Click the icon to see the alert:



If Cloudera Director is unable to connect to or upload usage information to the metering service, the alert will say:

- Cloudera Director is unable to send usage data to Cloudera’s billing service at <https://metering.cloudera.com>. Check that your network is configured to allow sending of usage data and that Cloudera’s billing service is running.

If Cloudera Director is unable to connect to Cloudera Manager, the alert will say, for example (with actual values for the names of your Cloudera Manager instance and environment, and time elapsed):

- Unable to connect to cm1 in env2 for at least 2 minutes 18 seconds. Check your deployment status. The deployment may have failed or may have a connectivity issue.

When an alert appears, check the network and security configuration where Cloudera Director is running:

- Check that the firewall rules for your Cloudera Director instance (for example, the security group for an AWS EC2 instance) are configured to permit network access to the internet.
- Check that the subnet for the Cloudera Director instance has a route to the internet.
- Check in the Cloudera Director web UI to ensure that Cloudera Director is able to connect to the Cloudera Manager instance.
- Open a shell on the Cloudera Director instance and try to ping a publicly-accessible URL, such as www.cloudera.com.
- Using a machine in your local network environment (outside of the network environment where Cloudera Director is running), send a ping request from a web browser to the collection service ping endpoint at this URL: <https://metering.cloudera.com/api/v1/ping>. If the metering service is not reachable, the service may be down. Contact Cloudera Support.

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.

Valid Role Types for Use in Configuration Files

For a list of valid roles for Cloudera Manager and CDH services that you can use in a Cloudera Director configuration file, see the Cloudera Manager API page on [Available Role Types](#).

Using Spot Instances

To help manage cloud resource costs, Cloudera supports Spot instances. Spot instances are Amazon EC2 instances that you can bid on. Unlike On-Demand Amazon EC2 instances, Spot instances only run as long as the price you bid exceeds the current Spot price. This allows you to add capacity to your workload at a low price.

Spot instances run just like On-Demand instances, except that they are not provisioned until the instance price falls below your bid. They also terminate automatically when the instance price exceeds or equals your bid price.

For more information about using Spot instances, see the [Amazon EC2 documentation](#). For help with bidding on Spot instances, see the [Spot Bid Advisor](#).

Planning for Spot Instances

It is normal for Spot instances on a cluster to disappear over time. However, Cloudera Manager does not see that these instances are terminated. If you restart a cluster that contains a Spot instance group, and the Spot instances have terminated, the restart fails. If you are modifying any group in the cluster that has lost Spot instances, do not select the **Restart** checkbox.

If your bid price is so low that you do not obtain an instance when the group is created, you will have 0 instances in your group. If this happens, you can:

- Delete the entire group.
- Add more instances to the group.
- Delete unprovisioned instances from the group (only as part of adding more instances to the group).
- Retry (repair) existing instances.

Customization and Advanced Configuration

You cannot do the following:

- Change the bid price
- Delete all instances without adding more

The bid price for Spot instances is set in an instance template. This template is associated with a group. Although you can modify the group, you cannot change the bid price. Therefore, if you set the bid price too low for successful provisioning, you must delete the group where that price is set and create a new group with the higher bid price. You must also delete the current group and create a new one if you want to drop the bid price.

Specifying Spot Instances

To specify Spot instances, create a new instance template and use this template for your group. For more information, see the steps for adding a cluster in the [Deploying Cloudera Manager and CDH on AWS](#) topic.

Best Practices for Using Spot Instances

- Use a Spot instance worker group in conjunction with an On-Demand worker group. This ensures that the cluster can redo computational tasks run on Spot instances that could be terminated before the tasks are finished.
- Use Spot instances only in contexts where the loss of the instance can be tolerated, as in a worker group. Do not use Spot instances for master nodes or for data storage.
- Use a minimum count of 0 for Spot instance groups. If you use a number above 0, the cluster will likely enter a failed state. If the cluster fails, contact Cloudera support for help.

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 5.5.1 on the AMI, the repository you specify should be `/5.5.1` and not `/5.5` 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](#).



Important: Cloudera Director supports only MBR and GPT partitions for AMIs that have a single partition on the root block device. AMIs with multiple partitions are not supported.

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.

Running Cloudera Director and Cloudera Manager in Different Regions or Clouds

A Cloudera Director instance requires network access to all of the Cloudera Manager and CDH instances it deploys and manages. If Cloudera Director is installed in the same subnet where you install Cloudera Manager and create CDH clusters, this requirement is satisfied automatically. However, the following alternative configurations are also supported:

- Running Cloudera Director in one region and Cloudera Manager and the CDH clusters it manages in a different region.
- Installing Cloudera Director on one cloud provider, such as AWS, and Cloudera Manager and the CDH clusters it manages on a different cloud provider, such as Google Cloud Platform.
- Installing Cloudera Director in your local network environment (on your laptop, for instance), and Cloudera Manager and the CDH clusters it manages in a cloud environment.

The most secure solution in these cases is to set up a VPN giving Cloudera Director access to the private subnet. Alternatively, Cloudera Director can be given SSH access to the instances through the public internet.

When using SSH to configure Cloudera Manager and CDH instances, Cloudera Director will try to connect to the instances in the following order:

1. Private IP address
2. Private DNS host name
3. Public IP address
4. Public DNS host name

The following requirements apply to running Cloudera Director and clusters in different regions or cloud provider environments when connecting to instances through their public endpoints:

- Your cluster instances must have public IP addresses and your security group must allow access to them through SSH.
- While Cloudera Director can run in a different subnet, Cloudera Manager and the CDH cluster hosts must be in the same subnet.
- Cloudera Director must have SSH access to the public IP addresses of all cluster instances.
- Cloudera Director needs to communicate with Cloudera Manager on its API endpoint (typically through HTTP to port 7189) on the private IP address. For security reasons, this endpoint should not be exposed to the public internet.
 - For Cloudera Manager instances that were deployed by Cloudera Director, if Cloudera Director cannot make a direct connection to the Cloudera Manager API on the private IP address, it will automatically attempt to create an SSH tunnel to the Cloudera Manager API endpoint through an SSH connection to the instance on its public IP address.
 - Connecting to an existing deployment of Cloudera Manager through SSH tunneling is not supported.

Deploying a Java 8 Cluster

When Cloudera Director installs Cloudera Manager and CDH clusters in the cloud, a version of the Java JDK is installed on each instance during the bootstrap process. By default, Cloudera Director installs a version of Java 7, but Java 8 can be installed, instead, by running the bootstrap script described on this page.

javaInstallationStrategy configuration

In order to use this bootstrap script, you must configure your deployment to use a `javaInstallationStrategy` value of `NONE`. This can be done using a configuration file or using the Cloudera Director API, but this property is not currently configurable in the Cloudera Director web UI. Here is how this setting would look in a configuration file:

```
...
cloudera-manager {
  instance: ${instances.m3x} {
    tags {
      application: "Cloudera Manager 5"
    }
  }
  javaInstallationStrategy: NONE
  ...
}
```

After the Cloudera Manager deployment has been created, additional Java 8 clusters can be added from the web UI using the bootstrap script.

Bootstrap script

The bootstrap script [java8-bootstrap-script.sh](#) is located on the Cloudera public GitHub site. Also on the site is a copy of the instructions for using the script, [Deploying a Java 8 cluster](#).

Use `java8-bootstrap-script.sh` as the bootstrap script for the instance templates in your cluster. This will install Java 8, which will be used to run Cloudera Manager and all of the cluster services. The following example shows how this might look in a configuration file:

```
instances {
  m3x {
    type: m3.xlarge
    image: ami-6283a827
    bootstrapScriptPath: "/script-path/java8-bootstrap-script.sh"
  }
}
```

Alternatively, you can copy the contents of the bootstrap script itself and use the `bootstrapScript` property instead.



Note: The URL in the script refers to CentOS/RHEL 7 and Cloudera Director 2.1.0. Update the URL to CentOS/RHEL 6 if necessary, depending on what operating system your cluster instances will run.

Creating AWS Identity and Access Management (IAM) Policies

In AWS, IAM files are used to create policies that control access to resources in a VPC. IAM roles allow EC2 instances to make API requests without the need to use or distribute AWS credentials (accessKey and secretAccessKey). For more information about IAM, see the [AWS Identity and Access Management User Guide](#) in the AWS documentation. For instructions on how to create an IAM role, see [Creating a Role to Delegate Permissions to an AWS Service](#) in the AWS documentation.

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
 - DescribeDBEngineVersions
- 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. For more information on ARNs, see [Amazon Resource Names \(ARNs\) and AWS Service Namespaces](#) in the AWS documentation.



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 CDH](#) on page 94.

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 83 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	<pre>\$ sudo yum install mysql-server</pre>
SLES	<pre>\$ sudo zypper install mysql</pre> <pre>\$ sudo zypper install libmysqlclient_r15</pre> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: Some SLES systems encounter errors with the <code>zypper install</code> command. For more information, see the Novell Knowledgebase topic, error running chkconfig. </div>
Ubuntu and Debian	<pre>\$ sudo apt-get install mysql-server</pre>

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	<pre>\$ sudo service mysqld stop</pre>
SLES, Ubuntu, and Debian	<pre>\$ sudo service mysql stop</pre>

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::

Customization and Advanced Configuration

- 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 higher. Comment out binlog_format for lower 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/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

5. 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.
6. 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>

7. Start the MySQL server:

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


8. 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>

OS	Command
	<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: 10px; margin-top: 10px;"> <p> 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 <code>Linux alternatives</code> command to set the system JDK to be the openJDK.</p> </div>
SLES	<code>\$ sudo zypper install mysql-connector-java</code>
Ubuntu or Debian	<code>\$ sudo apt-get install libmysql-java</code>

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, usernames, and passwords. Cloudera Director requires this information to connect to the database.

1. Log into MySQL as the root user:

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

2. 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 username (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:

```

Using MariaDB 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 CDH](#) on page 94.

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 MariaDB database instead of the embedded H2 database, as described below.

Installing the MariaDB Server



Note:

- If you already have a MariaDB database set up, you can skip to [Configuring and Starting the MariaDB Server](#) on page 87 to verify that your MariaDB 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 MariaDB database.

```
$ sudo yum install mariadb-server
```

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

Configuring and Starting the MariaDB Server

1. Stop the MariaDB server if it is running.

- For RHEL 6:

```
$ sudo service mariadb stop
```

- For RHEL 7:

```
$ sudo systemctl mariadb stop
```

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

3. 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 MariaDB to use the InnoDB engine, rather than MyISAM. (The default storage engine for MariaDB is MyISAM.) To check which engine your tables are using, run the following command from the MariaDB shell:

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

- To configure MariaDB 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 MariaDB 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.log
pid-file=/var/run/mysql/mysql.pid
```

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

- For RHEL 6:

```
$ sudo chkconfig mysqld on
```

- For RHEL 7:

```
$ sudo systemctl enable mariadb
```

6. Start the MariaDB server:

- For RHEL 6:

```
$ sudo service mysqld start
```

- For RHEL 7:

```
$ sudo systemctl mariadb start
```

7. Set the MariaDB 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 MariaDB JDBC Driver

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



Note: The JDBC driver described here to use for MariaDB is the MySQL driver, which works with MariaDB, as well.

1. Download the MySQL JDBC driver from <http://www.mysql.com/downloads/connector/j/5.1.html>.

Customization and Advanced Configuration

2. Extract the JDBC driver JAR file from the downloaded file. For example:

```
tar zxvf mysql-connector-java-5.1.31.tar.gz
```

3. Copy the JDBC driver, renamed, to the relevant host. For example:

```
$ sudo cp mysql-connector-java-5.1.31/mysql-connector-java-5.1.31-bin.jar  
/usr/share/java/mysql-connector-java.jar
```

If the target directory does not yet exist on this host, you can create it before copying the JAR file. For example:

```
$ sudo mkdir -p /usr/share/java/  
$ sudo cp mysql-connector-java-5.1.31/mysql-connector-java-5.1.31-bin.jar  
/usr/share/java/mysql-connector-java.jar
```



Note: Do not use the `yum install` command to install the MySQL driver package, because it installs openJDK, and then uses the Linux `alternatives` command to set the system JDK to be openJDK.

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, usernames, and passwords. Cloudera Director requires this information to connect to the database.

1. Log into MariaDB as the root user:

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

2. 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 MariaDB Database](#).

Backing Up MariaDB Databases

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

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

Configuring Cloudera Director Server to use the MariaDB 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 MariaDB 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 username (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 CDH cluster services.

Cipher Configuration

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

Table 2: Server Configuration Properties

Property	Description
lp.encryption.twoWayCipher	Cipher used to encrypt data. Possible values: <ul style="list-style-type: none"> desede - Triple DES (default) passthrough - No encryption transitional - Changing encryption
lp.encryption.twoWayCipherConfig	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 3: 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, Cloudera recommends 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, Cloudera recommends 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 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.encrypted.twoWayCipher` with the value `transitional`.
3. Configure `lp.encrypted.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;ZGVmYXVsdGRpcmVjdG9yZGVzZWR1a2V5|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.

Wait for the Server to Complete Ongoing Work

Do not try to change encryption while the server is performing ongoing work. 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. If this happens, you must configure the server for its old cipher, start it, and wait for that work to resume and be completed.

Changing from a Transitional Cipher to a Normal Cipher

After encryption has been changed using a transitional cipher, you can configure the server to use the new cipher normally.

Example: Assume the configuration string for the transitional cipher was as follows:

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

One restart of the server will suffice to pick up this change, and then the following configuration string for a normal cipher can be used:

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

Cloudera recommends that the server be left to run with a transitional cipher only until its next restart or upgrade, and then be reconfigured to use a normal cipher. There are two reasons for doing this:

- While configured with a transitional cipher, the server will not restart if work is waiting to be resumed.
- If the server is left configured with a transitional cipher, each time it is restarted the database contents will be re-encrypted using the same key.

Using an External Database for Cloudera Manager and CDH

By default, Cloudera Director configures Cloudera Manager and CDH services, such as Hive, to use the Cloudera Manager embedded PostgreSQL database. You can use Cloudera Director to configure them to use external database servers, instead, which is recommended for production environments. If you have a database server already configured, you can configure Cloudera Manager and CDH services to create or use databases on that server. You can also configure Cloudera Director to use a cloud provider service such as 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 for Cloudera Director server in one of the following ways:
 - Using the Cloudera Director web UI
 - Using the Cloudera Director REST API
 - By editing the `cluster.conf` file and launching the Cloudera Director server with the `bootstrap-remote` command

The topics in this section describe how to use Cloudera Director to define external database servers and external databases.

Defining External Database Servers

Cloudera Director needs information about external database servers before it can use them. This section describes defining database server templates and using Amazon Relational Database Service (RDS) to create new database servers..

The Database Server Template

A database server template can refer to either an existing database server or a server to be created. The 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, 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.



Note: A single database server is scoped to an environment, so only deployments and clusters in that environment recognize it.

Defining a Database Server Using the 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 <code>/databaseServers/</code>	Define a new database.	Admin required.
GET <code>/databaseServers/</code>	List all database servers.	
DELETE <code>/databaseServers/{name}</code>	Delete a database server definition.	Admin required.
PUT <code>/databaseServers/{name}</code>	Update a database server definition.	Admin required.
GET <code>/databaseServers/{name}</code>	Get a database server definition.	
GET <code>/databaseServers/{name}/status</code>	Get the status of a database server.	
GET <code>/databaseServers/{name}/template</code>	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.

Customization and Advanced Configuration

Defining a Database Server Using the Client Configuration File

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 {
  mysql1 {
    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 web UI and the server API.
- The database server must be in the same AWS region as Cloudera Director.

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 higher with the MySQL JDBC driver (used by Cloudera Director) version 5.1.19 or lower. 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
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: web UI

You can define an RDS database in AWS using the Cloudera Director web UI when you create a CM instance. In the Database Server section near the top of the Add Cloudera Manager wizard, click the dropdown list and select either **Create Database Server Instance** or **Register Existing Database Server**:

Database Server

Configurations (optional)

Select **Create Database Server Instance** to create a new MySQL database server with RDS. In the **Create Database Server Instance** window, enter credentials and configuration values for the database server:

Create Database Server Instance
✕

Name *

Master username

Master user password

DB type

Tags

Allocated storage (GB) *

Instance class *

DB subnet group name *

VPC security group IDs *

MySQL

?

?

?

?

?

?

?

?

?

➤ Advanced Options

Cancel

For more information about configuring a database in Amazon RDS see the [Amazon Relational Database Service Documentation](#).

Note: Cloudera Director also supports PostgreSQL database servers for Cloudera Manager and CDH, but they must be created outside of Cloudera Director and then treated as existing databases by selecting **Register Existing Database Server**.

Select **Register Existing Database Server** to use an existing MySQL or PostgreSQL database server. In the **Register Existing Database Server** window, enter information and credentials about your existing database server.

Register Existing Database Server



Name *	<input type="text"/>	
Hostname *	<input type="text"/>	
DB Port *	<input type="text"/>	
DB Username *	<input type="text"/>	
DB Password *	<input type="text"/>	
Type *	<input type="text" value="Please select a value"/>	

Cancel

OK

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 through 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 through GET.

Likewise, when the definition is deleted using 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 in AWS 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
    }
  }
}
```

Defining 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 instances 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

External Databases 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, indicated by the fact that it includes values for the hostnames or IP addresses and the ports.

```
cloudera-manager {
  # ...
  databases {
    CLOUDERA_MANAGER {
      name: scm1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: scmuser
      password: scmpassword
    }
    ACTIVITYMONITOR {
      name: am1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: amuser
      password: ampassword
    }
    REPORTSMANAGER {
      name: rm1
      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, which does not include hostnames or IP addresses and ports, 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: cmrmadmin
    }
    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 CDH 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/cloudera-director-client/application.properties` file (for the standalone client) or the `/etc/cloudera-director-server/application.properties` (for the server) in the Cloudera Director installation. The properties in these files take effect automatically. To override these properties, set new values in the command line.

For users upgrading Cloudera Director

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

- **RHEL and CentOS** - When new properties are introduced in Cloudera Director, they are added to `application.properties.rpmnew`. The original `application.properties` file functions as before and is not overwritten with the new Cloudera Director version properties. You do not need to copy the new properties from `application.properties.rpmnew` to the old `application.properties` file.
- **Ubuntu** - The modified Cloudera Director `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 properties. After upgrading, copy your changes from `application.properties.dpkg-old` to the new `application.properties` file.

All the new 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
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 Cloudera 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 Cloudera Director server access logging. Type: boolean Default: false
<code>lp.bootstrap.agents.maxNumberOfInstallAttempts</code>	Maximum number of times to retry installing Cloudera 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.parcel.distributeMaxConcurrentUploads</code>	Maximum concurrent uploads of parcels across cluster. Type: int Default: 5
<code>lp.bootstrap.parcel.distributeRateLimitKBs</code>	Maximum rate of parcel upload, in KB/s. Type: int Default: 256000

Property	Description
<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
<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

Property	Description
<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
<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

Property	Description
<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>
<code>lp.encryption.twoWayCipherConfig</code>	The configuration string for the chosen cipher. Type: string Default: <code>ZGVmYXVsdGRpcmVjdG9yZGVzZWR1a2V5</code> Cloudera recommends that you configure a different triple DES key. A warning appears in the server log if the default key is detected.
<code>lp.metrics.durationUnits</code>	Time units for reporting durations in metrics. Type: time unit Valid values: <code>DAYS</code> <code>HOURS</code> <code>MICROSECONDS</code> <code>MILLISECONDS</code> <code>MINUTES</code> <code>NANOSECONDS</code> <code>SECONDS</code> Default: <code>MILLISECONDS</code>
<code>lp.metrics.enabled</code>	Enable metrics gathering Type: boolean Default: false
<code>lp.metrics.location</code>	Directory for storing metrics reports. Type: directory Default: <code>\$LOG_DIR/metrics</code>
<code>lp.metrics.rateUnits</code>	Time units for reporting rates in metrics. Type: time unit Valid values: <code>DAYS</code> <code>HOURS</code> <code>MICROSECONDS</code> <code>MILLISECONDS</code> <code>MINUTES</code> <code>NANOSECONDS</code> <code>SECONDS</code> Default: <code>SECONDS</code>
<code>lp.metrics.reportingRate</code>	Frequency of metrics reporting, in minutes. Type: long Default: 1

Property	Description
<code>lp.pipeline.retry.maxNumberOfAttempts</code>	Maximum number of times to retry failed pipeline jobs; -1 = unlimited. Type: int Default: -1 for client, 16 for server
<code>lp.pipeline.retry.maxWaitBetweenAttempts</code>	Maximum wait time between pipeline retry attempts, in seconds. Type: int Default: 45
<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

Property	Description
	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 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

Property	Description
<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
<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 worker roles to restart at a time. Type: int Default: 20
<code>lp.update.restartCluster.rollingRestartSlaveFailCountThreshold</code>	Threshold for number of worker 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 worker 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

Property	Description
<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
<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.
- **Server** - Using the Cloudera Director web 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 web 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 web 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.

Cluster Configuration Using Cloudera Manager

Some configuration changes can safely be made to Cloudera Director-managed clusters using Cloudera Manager directly. For these use cases, Cloudera Director will sync up automatically with changes made in Cloudera Manager. Other configuration changes cannot be safely made using Cloudera Manager directly because Cloudera Director will not become aware of the change, resulting in failures when a user later tries to expand or otherwise modify the cluster.

For information on configuration changes and other changes to clusters that can and cannot be safely made directly through Cloudera Manager, see [Modifying or Updating Clusters Using Cloudera Manager](#) on page 116.

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 configuration file and the API.

Configuration File

Using the configuration file, 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 configuration file and the API.

Configuration File

Using the configuration file, 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 configuration file and the API.

Configuration File

Using the configuration file, 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.

Modifying or Updating Clusters Using Cloudera Manager

Some modifications or updates to a Cloudera Director-managed cluster can be made directly in Cloudera Manager. Other changes cannot be safely made directly in Cloudera Manager because Cloudera Director will not become aware of the change, resulting in failures when a user later tries to expand or otherwise modify the cluster.

The following table shows changes that are safe and unsafe to make directly in Cloudera Manager.

Description	Safe Changes	Unsafe Changes
Cloudera Manager or CDH version upgrade	Maintenance upgrades of Cloudera Manager or CDH, where only the 3rd digit of the Cloudera Manager or CDH version changes (for example, 5.4.0 to 5.4.3) are supported. The upgrade must be done using Cloudera Manager; it cannot be done from within Cloudera Director.	Minor or major version upgrades of Cloudera Manager or CDH (for example, 5.4 to 5.5 or 5 to 6) are not supported, even if done outside Cloudera Director.
Configuration changes	Configuration changes made on the Cloudera Manager Server or the Cloudera Management Service are supported if the configurations will not be affected by the addition of new	

Description	Safe Changes	Unsafe Changes
	<p>hosts to a cluster, that is, the configurations will not need to be set up on the new host.</p> <p>Configuration changes made to hosts, to CDH services and roles, or to the Cloudera Manager Server in areas that <i>will</i> impact future hosts (for example, parcels or agent-related configurations), are supported, but only in the present state of the deployment and cluster. You can use the cluster with these changes, but future modifications to the cluster from Cloudera Director will not propagate the changes to new hosts or roles you add to the cluster, since Cloudera Director will not be aware of the changes.</p>	
Role assignment		Adding new roles to hosts directly in Cloudera Manager is not supported.
Role migration		Migrating roles from one host to another within Cloudera Manager is not supported.
Decommissioning hosts outside Cloudera Director		Cloudera recommends that you do not decommission hosts using Cloudera Manager directly. Doing so puts Cloudera Director and Cloudera Manager out of sync with one another and can negatively impact future operations on the cluster.
Adding new hosts or clusters outside of Cloudera Director		Adding new hosts to existing clusters or adding a new cluster to Cloudera Manager, done directly in Cloudera Manager is not supported. Both these should be done from within Cloudera Director.
Changing Cloudera Manager username and password	This can be done in Cloudera Manager, but Cloudera Director has to be updated to be made aware of the new values.	
A Cloudera Manager cluster that was set up using Cloudera Manager, rather than through Cloudera Director		Not supported. Cloudera Director is not aware of clusters set up directly in Cloudera Manager.
Enabling Kerberos outside of Cloudera Director		Do not enable Kerberos through Cloudera Manager directly. Enable Kerberos using Cloudera Director 2.0 or higher. Use the configuration file, not the Cloudera Director web UI, to enable Kerberos.

Description	Safe Changes	Unsafe Changes
Modifying a cluster after enabling high availability outside Cloudera Director	If HDFS high availability has been enabled outside Cloudera Director, using Cloudera Manager directly, then you can run future Modify operations on the cluster, but only on those instance groups that do not contain highly available master roles on Cloudera Manager.	Do not enable high availability through Cloudera Manager directly, do so using Cloudera Director 2.0. Use the configuration file, not the Cloudera Director web UI, to enable high availability.
Upgrading your Cloudera Manager license	Upgrading a license using Cloudera Manager, for example, from Cloudera Express to Cloudera Enterprise, is supported.	

Post-Creation Scripts

Post-creation scripts are run after a Cloudera Manager cluster has been created. The scripts are run sequentially on a randomly selected cluster host. The scripts 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 client configuration file or the Cloudera Director API.

You can supply post-creation scripts in the client configuration file in two ways:

- Use the `postCreateScripts` directive inside of the `cluster {}` configuration block. This block can take an array of scripts, similar to the `bootstrapScript` that can be placed inside the `instance {}` configuration block.
- Use the `postCreateScriptsPaths` directive inside of the `cluster {}` configuration block. It can take an array of paths to arbitrary files on the local filesystem. This is similar to the `bootstrapScriptPath` directive. Cloudera Director reads the files from the filesystem and uses their contents as post-creation scripts.

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

Creating Kerberized Clusters With Cloudera Director

Using Cloudera Director 2.0 and higher with Cloudera Manager 5.5.0 and higher, you can create and configure Kerberized Cloudera Manager clusters. To launch a Kerberized cluster, edit the configuration file as described below and launch the cluster with Cloudera Director client, using the `bootstrap-remote` command to send the configuration file to a running Cloudera Director server.



Note: You must have an existing Kerberos Key Distribution Center (KDC) set up, and it must be reachable by the instance where Cloudera Director server is running and the instances where your Cloudera Manager cluster will be deployed. You must also set up a Kerberos realm for the cluster and a principal in that realm.



Important: Do not use Cloudera Manager to enable Kerberos on an existing cluster that is managed by Cloudera Director. Kerberos must be enabled through Cloudera Director using the configuration file.

Creating a Kerberized Cluster with the Cloudera Director Configuration File

A sample configuration file for creating Kerberized Cloudera Manager clusters is available on the Cloudera GitHub site: [director-scripts/kerberos/aws.kerberos.sample.conf](https://github.com/cloudera-director/director-scripts/kerberos/aws.kerberos.sample.conf).

The settings for enabling Kerberos are in the Cloudera Manager section of the configuration file. Provide values for the following configuration settings:

Configuration setting	Description
krbAdminUsername	An administrative Kerberos account with permissions that allow the creation of principals on the KDC that Cloudera Manager will be using. This is typically in the format <i>principal@your.KDC.realm</i>
krbAdminPassword	The password for the administrative Kerberos account.

Configuration setting	Description
KDC_TYPE	The type of KDC Cloudera Manager will use. Valid values are "MIT KDC" and "Active Directory".
KDC_HOST	The hostname or IP address of the KDC.
SECURITY_REALM	The security realm that the KDC uses.
AD_KDC_DOMAIN	The Active Directory KDC domain in the format of an X.500 Directory Specification (DC=domain,DC=example,DC=com). This setting is for Active Directory KDCs only.
KRB_MANAGE_KRB5_CONF	Set this to <code>true</code> . This allows Cloudera Manager to deploy Kerberos configurations to cluster instances. The value <code>false</code> is not supported for this configuration setting.
KRB_ENC_TYPES	The encryption types your KDC supports. Some of encryption types listed in the sample configuration file require the unlimited strength JCE policy files.

Other Kerberos configuration options are available to Cloudera Manager. For more information, see [Configuring Authentication](#) in the Cloudera Security guide.

The following example shows the `cloudera-manager` section of a configuration file with MIT KDC Kerberos enabled:

```
cloudera-manager {
  instance: ${instances.cm-image} {
    tags {
      application: "Cloudera Manager 5"
    }
  }
}

#
# Automatically activate 60-Day Cloudera Enterprise Trial
#
enableEnterpriseTrial: true

unlimitedJce: true
# Kerberos principal and password for use by Cloudera Director
krbAdminUsername: "principal@my.kdc.realm"
krbAdminPassword: "password"

# Cloudera Manager configuration values
configs {
  CLOUDERA_MANAGER {
    KDC_TYPE: "MIT KDC"
    KDC_HOST: "KDC_host_ip_address"
    SECURITY_REALM: "my_security_realm"
    KRB_MANAGE_KRB5_CONF: true
    KRB_ENC_TYPES: "aes256-cts aes128-cts des3-hmac-sha1 arcfour-hmac des-hmac-sha1
des-cbc-md5 des-cbc-crc"
  }
}
}
```

Creating Highly Available Clusters With Cloudera Director

Using Cloudera Director 2.0 or higher and Cloudera Manager 5.5 or higher, you can launch highly available clusters for HDFS, YARN, ZooKeeper, HBase, Hive, Hue, and Oozie. The services are highly available on cluster launch with no additional setup. To enable high availability, edit the Cloudera Director configuration file as described in this topic and launch the cluster with the Cloudera Director client and the `bootstrap-remote` command, which sends the configuration file to a running Cloudera Director server.



Note: With Cloudera Director 1.5 and Cloudera Manager 5.4, you can set up a highly available cluster by running a script after the cluster is launched. For more information, see the [high-availability scripts](#) and the [README file](#) on the [Cloudera Director GitHub site](#).

Limitations and Restrictions

The following limitations and restrictions apply to creating highly available clusters with Cloudera Director:

- The procedure described in this section works with Cloudera Director 2.0 or higher and Cloudera Manager 5.5 or higher.
- Cloudera Director does not support migrating a cluster from a non-high availability setup to a high availability setup.
- Cloudera recommends sizing the master nodes large enough to support the desired final cluster size.
- Settings must comply with the configuration requirements described below and in the `aws.ha.reference.conf` file. Incorrect configurations can result in failures during initial bootstrap.

Editing the Configuration File to Launch a Highly Available Cluster

Follow these steps to create a configuration file for launching a highly available cluster.

1. Download the sample configuration file `aws.ha.reference.conf` from the Cloudera GitHub site. The cluster section of the file shows the role assignments and required configurations for the services where high availability is supported. The file includes comments that explain the configurations and requirements.
2. Copy the sample file to your home directory before editing it. Rename the `aws.ha.reference.conf` file, for example, to `ha.cluster.conf`. The configuration file must use the `.conf` file extension. Open the configuration file with a text editor.



Note: The sample configuration file includes configuration specific to Amazon Web Services, such as the section for cloud provider credentials. The file can be modified for other cloud providers by copying sections from the other cloud provider-specific sample files, for example, [gcp.simple.conf](#).

3. Edit the file to supply your cloud provider credentials and other details about the cluster. A highly available cluster has additional requirements, as seen in the sample `aws.ha.reference.conf` file. These requirements include duplicating the master roles for highly available services.

The sample configuration file includes a set of instance groups for the services where high availability is supported. An instance group specifies the set of roles that are installed together on an instance in the cluster. The master roles in the sample `aws.ha.reference.conf` file are included in four instance groups, each containing particular roles. The names of the instance groups are arbitrary, but the names used in the sample file are `hdfs masters-1`, `hdfs masters-2`, `masters-1`, and `masters-2`. You can create multiple instances in the cluster by setting the value of the `count` field for the instance group. The sample file is configured for two `hdfs masters-1` instances, one `hdfs masters-2` instance, two `masters-1` instances, and one `masters-2` instance.

The cluster services for which high availability is supported are listed below, with the minimum number of roles required and other requirements.

- HDFS
 - Two NAMENODE roles.
 - Three JOURNALNODE roles.
 - Two FAILOVERCONTROLLER roles, each colocated to run on the same host as one of the NAMENODE roles (that is, included in the same instance group).
 - One HTTPFS role if the cluster contains a Hue service.
 - The NAMENODE `nameservice`, `autofailover`, and `quorum journal name` must be configured for high availability exactly as shown in the sample `aws.ha.reference.conf` file.

- Set the HDFS service-level configuration for fencing as shown in the sample `aws.ha.reference.conf` file:

```
configs {
    # HDFS fencing should be set to true for HA configurations
    HDFS {
        dfs_ha_fencing_methods: "shell(true)"
    }
}
```

- Three role instances are required for the HDFS JOURNALNODE role. This ensures a quorum for determining which is the active node and which are standbys.

For more information, see [HDFS High Availability](#) in the Cloudera Administration documentation.

- YARN

- Two RESOURCEMANAGER roles.
- One JOBHISTORY role.

For more information, see [YARN \(MRv2\) ResourceManager High Availability](#) in the Cloudera Administration documentation.

- ZooKeeper

- Three SERVER roles (recommended). There must be an odd number, but one will not provide high availability
- Three role instances are required for the ZooKeeper SERVER role. This ensures a quorum for determining which is the active node and which are standbys.

- HBase

- Two MASTER roles.
- Two HBASETHRIFTSERVER roles (needed for Hue).

For more information, see [HBase High Availability](#) in the Cloudera Administration documentation.

- Hive

- Two HIVESERVER2 roles.
- Two HIVEMETASTORE roles.

For more information, see [Hive Metastore High Availability](#) in the Cloudera Administration documentation.

- Hue

- Two HUESERVER roles.
- One HTTPFS role for the HDFS service.

For more information, see [Hue High Availability](#) in the Cloudera Administration documentation.

- Oozie

- Two SERVER roles.
- Oozie plug-ins must be configured for high availability exactly as shown in the sample `aws.ha.reference.conf` file. In addition to the required Oozie plug-ins, other Oozie plug-ins can be enabled. All Oozie plug-ins must be configured for high availability.
- Oozie requires a load balancer for high availability. Cloudera Director does not create or manage the load balancer. The load balancer must be configured with the IP addresses of the Oozie servers after the cluster completes bootstrapping.

For more information, see [Oozie High Availability](#) in the Cloudera Administration documentation.

- The following requirements apply to databases for your cluster:

- You can configure external databases for use by the services in your cluster and for Cloudera Director. If no databases are specified in the configuration file, an embedded PostgreSQL database is used.

- External databases can be set up by Cloudera Director, or you can configure preexisting external databases to be used. Databases set up by Cloudera Director are specified in the `databaseTemplates` block of the configuration file. Preexisting databases are specified in the `databases` block of the configuration file. External databases for the cluster must be either all preexisting databases or all databases set up by Cloudera Director; a combination of these is not supported.
- Hue, Oozie, and the Hive metastore each require a database.
- Databases for highly available Hue, Oozie, and Hive services must themselves be highly available. An Amazon RDS MySQL Multi-AZ deployment, whether preexisting or configured to be created by Cloudera Director, satisfies this requirement.

Using Role Migration to Repair HDFS Master Role Instances


Cloudera Director supports exact one-for-one host replacement for HDFS master role instances. This is a partially manual process that requires migration of the roles in Cloudera Manager. If a host running HDFS master roles (NameNode, Failover Controller, and JournalNode) fails in a highly available cluster, you can use Cloudera Director and the Cloudera Manager Role Migration wizard to move the roles to another host without losing the role states, if any. The previously standby instance of each migrated role runs as the active instance. When the migration is completed, the role that runs on the new host becomes the standby instance.

Keep in mind the following when performing HDFS role migration:

- Do not modify any instance groups on the cluster during the repair and role migration process.
- Do not clone the cluster during the repair and role migration process.
- To complete the migration (Step 3 below), click a checkbox to indicate that the migration is done, after which the old instance is terminated. Check Cloudera Manager to ensure that the old host has no roles or data on it before performing this step in Cloudera Director. Once the old instance is terminated, any information or state it contained is lost.
- If you have completed Step 1 (on Cloudera Director) and intend to complete Step 2 (on Cloudera Manager) at a later time, you can confirm which IP address to migrate from or to by going to the cluster status page in Cloudera Director and clicking either the link for migration in the upper left, or the **Modify Cluster** button on the right. A popup displays the hosts to migrate from and to:

Manual Role Migration

Attention! The following instances have master roles that need to be manually migrated from within Cloudera Manager:

Migrate the roles by using Cloudera Manager commands 

I have manually migrated these roles

Original Instance	New Instance	Roles to Migrate
178.28.5.37	178.28.4.199	ZooKeeper: Server HDFS: NameNode, Failover Controller, JournalNode

Ignore for now, I will complete manual role migration later

OK

- You do not need to check the boxes to restart and deploy client configuration at the start of the repair process. You restart and deploy the client configuration manually after role migration is complete.
- Do not attempt repair for non-highly available master roles. The Cloudera Manager Role Migration wizard only works for high availability HDFS roles.

Step 1: In Cloudera Director, Create a New Instance

1. In Cloudera Director, click the cluster name and click **Modify Cluster**.

Customization and Advanced Configuration

2. Click the checkbox next to the IP address of the failed instance (containing the HDFS NameNode and colocated Failover Controller, and possibly a JournalNode). Click **Repair**.
3. Click **OK**. You do not need to select **Restart Cluster** at this time, because you will restart the cluster after migrating the HDFS master roles.

Cloudera Director creates a new instance on a new host, installs the Cloudera Manager agent on the instance, and copies the Cloudera Manager parcels to it.

Step 2: In Cloudera Manager, Migrate Roles and Data

Open the cluster in Cloudera Manager. On the **Hosts** tab, you see a new instance with no roles. The cluster is in an intermediate state, containing the new host to which the roles will be migrated and the old host from which the roles will be migrated.

Use the Cloudera Manager **Migrate Roles** wizard to move the roles.

See [Moving Highly Available NameNode, Failover Controller, and JournalNode Roles Using the Migrate Roles Wizard](#) in the Cloudera Administration guide.

Step 3: In Cloudera Director, Delete the Old Instance

1. Return to the cluster in Cloudera Director.
2. Click **Details**. The message "Attention: Cluster requires manual role migration" is displayed. Click **More Details**.
3. Check the box labeled, "I have manually migrated these roles."
4. Click **OK**.

The failed instance is deleted from the cluster.

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

For this method, you use the Cloudera Director client and the `bootstrap-remote` command to send a configuration file to the Cloudera Director server 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>
    }
  }
}
```

If you don't include an entry for Sentry in the `databases` section of the configuration file, the Cloudera Director default database, PostgreSQL, will be used, rather than the Cloudera Manager default database for Sentry, which is MySQL.

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.

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](#).

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 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 web 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 web 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 or Google Cloud Platform instance metadata or for service or role configurations, special characters, such as periods and colons, must be enclosed in double quotes. This includes some characters required by the HOCON format. For example, a tag value that would require quoting is "company:department:team". See the AWS and Google Cloud Platform 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 137.
2. Install the latest version of the Cloudera Director client from the [Cloudera Director Download Page](#).
3. Enter the following command:

```
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 hostname or IP address of the instance on which Cloudera Director server is running, and *port* is the port on which it is listening. The default port for Cloudera Director is 7189.

Both the Cloudera Director client (in the terminal where the `bootstrap-remote` command was issued) and the Cloudera Director server web UI display the status throughout the deployment process.

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.
6. Enter the version of CDH to deploy in the **Version** field or leave the default value. By default, the version of CDH that will be installed depends on the version of Cloudera Director you are using:
 - If you are using Cloudera Director 2.0, the latest released version of Cloudera Manager/CDH 5.5 will be installed by default.
 - If you are using Cloudera Director 2.1, the latest released version of Cloudera Manager/CDH 5.7 will be installed by default.

To install an earlier or later version of CDH than the default version, perform the following steps:

- a. Enter the desired CDH version in the **Version** field of the **Products** section. For example, for CDH 5.4.8 enter 5.4.8.
- b. Scroll down to **Configurations (optional)** and expand the section.
- c. Click **Override default parcel repositories**.
- d. Enter the repository parcel URL for the version of CDH you want to install. Parcel URLs for versions of CDH 5 take the form <http://archive.cloudera.com/cdh5/parcels/>, followed by the major, minor, and (if applicable) dot release number. For example, the URL for CDH 5.4.8 is <http://archive.cloudera.com/cdh5/parcels/5.4.8>.



Note: The CDH minor version must not be greater than the Cloudera Manager minor version. For example, CDH 5.7 will not work with Cloudera Manager 5.5, but CDH 5.7 (or lower) will work with Cloudera Manager 5.7.

7. Select the type of cluster to deploy from **Services**.
8. Select the numbers of masters, workers, and gateways to deploy. Then, select an instance template for each or create one or more new templates.
9. 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.numberOfWorkingCacheExecutorThreads` - 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 web 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.

Creating and Modifying Clusters with the Cloudera Director web UI

Before initially launching a CDH cluster, you can use the Cloudera Director web UI to add, delete, or modify the default roles and instance groups. You can also add, remove, or repair instances in an existing cluster.

Configuring Instance Groups During Cluster Creation

An *instance group* is a collection of roles that are installed together on one or more instances. When Cloudera Director creates a Cloudera Manager cluster, it includes three default instance groups: masters, workers, and gateway. Each of these instance groups contains roles of the type represented by that instance group, for the CDH services selected for the cluster. For example, if your cluster includes HDFS and YARN, the masters instance group includes the following roles:

- For HDFS - NameNode, SecondaryNameNode, Balancer
- For YARN - ResourceManager, JobHistory Server

The workers instance group will include the following roles:

- For HDFS - DataNode
- For YARN - NodeManager

The gateway instance group includes a gateway role for HDFS and another for YARN.

For an introduction to master, worker, and gateway roles, see the [Cloudera Manager 5 Overview](#).

Although the default instance groups are automatically configured with roles of a given type (masters, workers, or gateway), you can add any kind of role to any instance group.

When you create a cluster with Cloudera Director, a default set of instance groups and roles, based on the CDH services you include, is displayed in the Instance Groups section of the Add Cluster page:

Instance groups

Name ?	Roles	Instance Template	Instance Count	
masters	Edit Roles	Select a Temp ▾	1 ↕	Delete Group
workers	Edit Roles	Select a Temp ▾	10 ↕	Delete Group
gateway	Edit Roles	Select a Temp ▾	1 ↕	Delete Group
Add Group				

By clicking **Edit Roles**, you can see the roles included in each instance group. These roles will be installed on each instance running that instance group. In this example, by clicking **Edit Roles** for the workers instance group above, you can see that each of the 10 instances that will be installed for the workers instance group will include two roles, an HDFS DataNode and a YARN NodeManager:

Instance group: workers ✕

Role Assignment	Service	Role
	HDFS	Add Role ▾ DataNode x
	Hive	Add Role ▾
	Hue	Add Role ▾
	Oozie	Add Role ▾
	Sqoop 2	Add Role ▾
	YARN	Add Role ▾ NodeManager x
	ZooKeeper	Add Role ▾

Cancel
Reset
OK

You can modify the default configuration of instance groups during cluster creation by doing the following:

- Change the number of instances for an instance group by clicking the up or down arrows.
- Delete an instance group by clicking **Delete Group** at the right end of the row for that instance group.
- Add roles to an existing instance group by clicking **Edit Roles** and then **Add Role**. Available roles for the services in the cluster are displayed. Click a role to add it to the instance group.
- Add another instance group to the cluster by clicking **Add Group**, entering a name for the instance group and assigning roles to it, selecting an instance template, and clicking the up or down arrows to choose the number of instances to install.

Modifying the Number of Instances in an Existing Cluster

Cloudera Director can grow or shrink the size of an existing cluster by adding or removing instances.

Adding Instances to a Cluster

1. Log in to Cloudera Director at `http://director-server-hostname:7189`. Cloudera Director opens on the All Environments page, which displays the current environments, deployments, and clusters. Click the cluster you want to modify.
2. Click **Modify Cluster** to the right of the cluster name. The Modify Cluster page displays the gateway, masters, and workers instance groups and any additional instance groups that have been added to the cluster, with the current number of instances in each instance group.
3. You can add instances to an existing instance group or create a new instance group and add roles to it.
 - To add instances to an existing instance group, click **Edit** to the right of the instance group and click the up or down arrows in the **Add Instances** section to increase the number of workers and gateways to the desired size. Each new instance will contain the same roles as the existing instances of that group.
 - To create a new instance group, click **Add Group**, enter a name for the instance group, assign roles to it, select an instance template, and click the up or down arrows to choose the desired number of instances of that group to add.



Note: Cloudera recommends rebalancing the cluster through Cloudera Manager if you increase the number of HDFS DataNodes by 30% or more. For more information, see [Rebalancing the Cluster After Adding or Removing Instances](#) on page 132.

Removing Instances from a Cluster

1. Log in to Cloudera Director at `http://director-server-hostname:7189`.
Cloudera Director opens on the All Environments page, which displays the current environments, deployments, and clusters. Click the cluster you want to modify.
2. Click **Modify Cluster** to the right of the cluster name. The Modify Cluster page displays the gateway, masters, and workers instance groups and any additional instance groups that have been added to the cluster, with the current number of instances in each instance group.
3. You can remove an entire instance group, including all of its instances, or remove individual instances from an instance group:
 - To remove an entire instance group, click **Delete Group** at the right end of the row for that instance group.
 - To remove individual instances from an instance group, click **Edit** near the right end of the row for the instance group. Click the checkbox for each instance you want to remove, and click the **Delete** button. The instances you select 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 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.
- Cloudera Director decommissions instances before removing them from the cluster. When decommissioning an HDFS DataNode, Cloudera Manager moves all the blocks from that instance to other instances so that the replication factor is maintained, and there is no risk of data loss.
- You cannot delete an instance with an HDFS DataNode if the number of DataNodes equals the replication factor (which by default is three) of any file stored in HDFS. For example, if the replication factor of any file is three, and you have three DataNodes, you cannot delete an instance with a DataNode.
- Cloudera recommends rebalancing the cluster through Cloudera Manager if you reduce the number of HDFS DataNodes by 30% or more. For more information, see [Rebalancing the Cluster After Adding or Removing Instances](#) on page 132.

Rebalancing the Cluster After Adding or Removing Instances

After you add or remove instances from a cluster, HDFS data is likely to be distributed unevenly across DataNodes. Cloudera Director does not rebalance HDFS when you add instances or remove them from the cluster. If you need to rebalance the cluster, you must do so manually as described in [HDFS Balancers](#) in the Cloudera Manager documentation.

The need for rebalancing depends on the amount of data in HDFS and the number of instances added or removed during the cluster. Rebalancing is required only when there is a large movement of data. Cloudera recommends rebalancing the cluster through Cloudera Manager if you increase or reduce the number of DataNodes by 30% or more.

Repairing Worker and Gateway Instances in a Cluster

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

Cloudera Director opens on the All Environments page, which displays the current environments, deployments, and clusters. Click the cluster you want to modify.

2. Click **Modify Cluster** to the right of the cluster name. The Modify Cluster page displays the gateway, masters, and workers instance groups and any additional instance groups that have been added to the cluster, with the current number of instances in each instance group.
3. Click **Edit** next to the instance count for workers or gateways to repair, and select the instances to repair.
4. Click the **Repair** button above the list of instances. The instances 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 without making any changes.
6. Click **Continue** to confirm and repair the selected instances.



Note: The above procedure is for worker and gateway roles, not for master roles. Because master roles have state, repairing them requires migrating the roles from one host to another. For information on migrating HDFS master roles, see [Using Role Migration to Repair HDFS Master Role Instances](#) on page 123.

Terminating a Cluster

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

Terminating a Cluster with the web UI

To terminate a cluster with the web 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 web UI or the API.

Managing Users with the Cloudera Director Web web UI

You can perform the following user management operations through the Cloudera Director Web 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**.

Managing Cloudera Manager Instances with Cloudera Director Server

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 as unavailable 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.

REST method	Description
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.

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**.

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 in standalone mode, without Cloudera Director server, perform the tasks below. You must be either running as root or using sudo to perform these tasks.

For instructions on installing Cloudera Director client together with Cloudera Director server, see the following:

- For AWS, see [Installing Cloudera Director Server and Client on the EC2 Instance](#) on page 27.
- For Google Cloud Platform, see [Installing Cloudera Director Server and Client on Google Compute Engine](#) on page 40.



Important: Cloudera Director requires a JDK. For more information, see [Supported Software and Distributions](#) on page 20.

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

- For RHEL 6 and CentOS 6:

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

- For RHEL 7 and CentOS 7:

```
cd /etc/yum.repos.d/
sudo wget "http://archive.cloudera.com/director/redhat/7/x86_64/director/cloudera-director.repo"
```

- For Ubuntu 14.04 (Trusty Tahr):

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

3. Add the signing key.

- For RHEL 6, CentOS 6 this step is not required. Continue to the next step.
- For RHEL 7, CentOS 7 this step is not required. Continue to the next step.
- 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 and CentOS 6:

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


- For RHEL 7 and CentOS 7:

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

- For Ubuntu 14.04 (Trusty Tahr):

```
apt-get install 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:
 - a. 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.
 - b. 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 web 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).

c. 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 web 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 web 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 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
bootstrap	Bootstraps an environment, deployment, and cluster locally (in standalone mode).	<code>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>.
status	Reports status on various entities, including deployment, cluster,	

Command	Description	Options
	Cloudera Manager instance, and cluster instances.	
terminate	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>.
update	Updates an environment, deployment, and cluster locally (in standalone mode).	
validate	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
bootstrap-remote	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> <code>lp.remote.username=Cloudera Director server username</code> <code>lp.remote.password=Cloudera Director server password</code>
terminate-remote	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 with Cloudera Director Client

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 Director 2.0.0 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 the Configuration File

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 with the Configuration File

After launching a cluster with the `bootstrap` command (using the stand-alone Cloudera Director client), you can add or remove instances with the `update` command:

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.



Note: If you create a cluster with Cloudera Director server using the `bootstrap remote` command, you cannot modify the cluster with the CLI, but only with the Cloudera Director web UI.

Rebalancing the Cluster After Adding or Removing Hosts

After hosts have been added to or removed from a cluster, HDFS data is likely to be distributed unevenly across DataNodes. Cloudera Director does not rebalance HDFS when you add hosts or remove them from the cluster, so after growing or shrinking the cluster, you must perform manual rebalances in Cloudera Manager, as described in the Cloudera Manager documentation, [HDFS Balancers](#).

The need for rebalancing depends on the amount of data in HDFS and the number of hosts added or removed during the cluster. Cloudera Director decommissions hosts before removing them from the cluster during a shrink operation. As part of decommissioning a DataNode, Cloudera Manager will move all the blocks from that host to other hosts so that the replication factor will be maintained even after the hosts are decommissioned. So there is no risk of data loss if the cluster is shrunk by more than two instances at a time. Rebalancing is necessary so that the blocks are placed in an optimal manner and is not required when a small number of hosts have been removed from a cluster, but only when there has been a large movement of data.

Upgrading Cloudera Director

This section contains notes and procedures for upgrading Cloudera Director.

Before Upgrading Cloudera Director

Follow these steps before upgrading Cloudera Director.

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 web UI), an upgrade will not complete successfully. An error in the log file instructs you 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 at `/var/lib/cloudera-director-server/state.h2.db`.

If you are using a MySQL database to store the 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. If upgrading from Cloudera Director 1.1, change your default encryption key.

After an upgrade from Cloudera Director 1.1 to a higher version, 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 after performing an upgrade from 1.1 to a higher version. 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 91.

Changes to the `application.properties` File

If you modified your existing `application.properties` file, the result of upgrading depends on which version of Linux you are using:

- **RHEL and CentOS** - When new properties are introduced in Cloudera Director, they are added to `application.properties.rpmnew`. The original `application.properties` file functions as before and is not overwritten with the new Cloudera Director version properties. You do not need to copy the new properties from `application.properties.rpmnew` to the old `application.properties` file.
- **Ubuntu** - The modified Cloudera Director `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 properties. After upgrading, copy your changes from `application.properties.dpkg-old` to the new `application.properties` file.

Requirements for Cloudera Director 2.0 and Higher

The following are requirements for running Cloudera Director 2.0 and higher:

- Cloudera Director 2.0 and higher support the following Linux operating systems:
 - RHEL and CentOS 6.5, 6.7, and 7.1
 - Ubuntu 14.04

Upgrading Cloudera Director

- Cloudera Director now requires Oracle JDK (Oracle Java SE Development Kit) version 7 or 8. Java 6 is not supported.
- Cloudera Director 2.0 and higher can install any version of Cloudera Manager 5 with any CDH 5 parcels. Cloudera Manager 4 and CDH 4 are not supported. Use of CDH packages is not supported.

If you are running a lower version of Cloudera Director on an operating system that is not supported for Cloudera Director 2.0, you cannot upgrade to Cloudera Director 2.0 or higher.

For complete requirements for Cloudera Director, see [Requirements and Supported Versions](#).

Changes in Cloudera Director 2.0

- Cloudera Director now requires Oracle JDK (Oracle Java SE Development Kit) version 7 or 8. Java 6 is not supported.
- Cloudera Director 2.0 can install any version of Cloudera Manager 5 with any CDH 5 parcels. Cloudera Manager 4 and CDH 4 are not supported. Use of CDH packages is not supported.

Handling Modified Plug-in Configuration Files

Cloudera Director includes plug-in configuration files that enable you to configure how the plug-ins work. The following plug-in files are located in directories in `/var/lib/cloudera-director-plugins/`:

- `aws-provider-version`
- `azure-provider-version`
- `byon-provider-example-version`
- `google-provider-version`
- `sandbox-provider-version`

The location for plug-in configuration files has changed starting with Cloudera Director 2.0. In Cloudera Director 1.5.x and lower, they are located at `/var/lib/cloudera-director-server/plugins`. In Cloudera Director 2.0 and higher, they are located at `/var/lib/cloudera-director-plugins/`.

You do not normally need to modify the plug-in configuration files, but if you have modified any of them, your modifications will be overwritten during an upgrade. Before running the `upgrade` command, back up the modified files to another location. Then, after upgrading, redo your modifications in the new version of the file. These steps are included in the upgrade procedures below.

Upgrading Cloudera Director

The following sections describe steps for upgrading Cloudera Director 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. Cloudera Director 2.1.x requires Java 7 or 8. If you must upgrade your version of the Java SDK to meet this requirement, do so now.
3. Update your Cloudera Director `.repo` file (the yum repository configuration file) to point to the version of Cloudera Director you are upgrading to by doing one of the following:
 - Open `/etc/yum.repos.d/cloudera-director.repo`. The `baseurl` value in this file now points to your current version of Cloudera Director, such as `/1` or `/2` (and may include a specific minor or maintenance release version, such as `/1.1`, `/1.1.3`, `/2.0`, or `/2.0.0`). Update the `baseurl` value to point to the new version, `/2`.



Note: Cloudera software version numbers take the form *major_release.minor_release.maintenance_release*. If there is no major or minor release number, as in /2, the latest version of 2.x is used.

In the absence of a minor version

- Instead of editing your existing `.repo` file, you can download a new Cloudera Director `.repo` file, which will point to the latest version of Cloudera Director:

```
cd /etc/yum.repos.d/
sudo wget "http://archive.cloudera.com/director/redhat/7/x86_64/director/cloudera-director.repo"
```

To upgrade to a version of Cloudera Director other than the latest version, you can edit the newly downloaded `.repo` file as described in the previous bullet point.

4. If you have not modified the plug-in configuration files, skip to the next step. If you modified any configuration files in `/var/lib/cloudera-director-plugins/plug-in_name-version` (or, for Cloudera Director 1.5 or lower, in `/var/lib/cloudera-director-server/plugins/plug-in_name-version`), back them up to another location and remove them from this location before running the upgrade command.
5. Issue the following commands:

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

6. If you have not modified any configuration files, skip to the next step. If you modified any configuration files, restore your backed up files now to `/var/lib/cloudera-director-plugins/plug-in_name-new_version`, before restarting the Cloudera Director server.
7. Restart the Cloudera Director server:

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



Note: Installing the Cloudera Director server and client packages will automatically install the required plug-in package.

Ubuntu

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

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

2. Cloudera Director 2.1.x requires Java 7 or 8. If you must upgrade your version of the Java SDK to meet this requirement, do so now.
3. Update your Cloudera Director `cloudera-director.list` file (the repository configuration file) to point to the version of Cloudera Director you are upgrading to by doing one of the following:
 - Open `/etc/apt/sources.list.d/cloudera-director.list`. The `baseurl` value in this file now points to your current version of Cloudera Director, such as `trusty-director1` or `trusty-director2` (and may include a specific minor or maintenance release version, such as `trusty-director1.1`, `trusty-director1.1.3`, `trusty-director2.0`, or `trusty-director2.0.0`). Update the `baseurl` value to point to the newest version, `trusty-director2`, if this is not already the current value.



Note: Cloudera software version numbers take the form *major_release.minor_release.maintenance_release*. If there is no major or minor release number, as in `trusty-director2`, the latest version of 2.x is used.

- Instead of editing your existing `cloudera-director.list` file, you can download a new Cloudera Director `cloudera-director.list` file, which will point to the latest version of Cloudera Director:

```
cd /etc/apt/sources.list.d/  
sudo curl "http://archive.cloudera.com/director/ubuntu/trusty/amd64/director/cloudera-director.list"
```

To upgrade to a version of Cloudera Director other than the latest version, you can edit the newly downloaded `cloudera-director.list` file as described in the previous bullet point.

4. If you have not modified the plug-in configuration files, skip to the next step. If you modified any configuration files in `/var/lib/cloudera-director-plugins/plug-in_name-version` (or, for Cloudera Director 1.5 or lower, in `/var/lib/cloudera-director-server/plugins/plug-in_name-version`), back them up to another location and remove them from this location before running the upgrade command.
5. 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
```

6. If your original Cloudera Director `application.properties` file has not been modified, proceed to the next step. If your `application.properties` file was modified, the original properties file will be overwritten by the new properties file containing new Cloudera Director properties, as described above in [Changes to the application.properties File](#) on page 143. Copy your changes from `application.properties.dpkg-old` to the new `application.properties` file before restarting the server.
7. If you have not modified any configuration files, skip to the next step. If you modified any configuration files, restore your backed up files now to `/var/lib/cloudera-director-plugins/plug-in_name-new_version`, before restarting the Cloudera Director server.
8. Restart the Cloudera Director server:

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



Note: Installing the Cloudera Director server and client packages will automatically install the required plug-in package.

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 81. 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.

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
```

Backing Up the H2 Embedded Database

By default, 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
```

Back up the `state.h2.db` file to avoid losing environment and cluster data. To ensure that your backup copy can be restored, you should use the H2 backup tools and rather than simply copying the file. For more information, see the [H2 Tutorial](#).

Cloudera Director Cannot Manage a Cluster That Was Kerberized Through Cloudera Manager

Symptom

Cloudera Director cannot manage a cluster after Cloudera Manager is used to enable Kerberos on the cluster.

Cause

Once a cluster is deployed through Cloudera Director, some changes to the cluster that are made using Cloudera Manager cause Cloudera Director to be out of sync, and hence unable to manage the cluster. See [Modifying or Updating Clusters Using Cloudera Manager](#).

Solution

Deploy a new kerberized cluster, use `distcp` to transfer data from the old cluster to the new one, and then destroy the old cluster.

New Cluster Fails to Start Because of Missing Roles

Symptom

A new cluster will not start because roles are missing.

Cause

Cloudera Director does not validate that all necessary roles are assigned when provisioning a cluster. This can lead to failures during the initial run of a new cluster. For example, if the gateway instance group was removed but the Flume Agent and Kafka Broker were assigned to roles in that group, the cluster will fail to start.

Solution

Ensure that all required role types for the CDH services included in the cluster are assigned to instances before starting the cluster.

Cloudera Director Server Will Not Start with Unsupported Java Version

Symptom

Cloudera Director server will not start, and `/var/log/cloudera-director-server/cloudera-director-server.out` has the following error:

```
Exception in thread "main" java.lang.UnsupportedClassVersionError:  
com/cloudera/launchpad/Server : Unsupported major.minor version 51.0
```

Cause

You are running Cloudera Director server against an older, unsupported version of the Oracle Java SE Development Kit (JDK).

Solution

Update to Oracle JDK version 7 or 8.

Error Occurs if Tags Contain Unquoted Special Characters

Symptom

When using the configuration file with the `bootstrap` command to start Cloudera Director client, or using the `bootstrap-remote` command to set up a cluster with Cloudera Director server, an error message is displayed. 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"`.

```
com.typesafe.config.ConfigException$WrongType: ... <x> has type OBJECT rather than STRING
```

Cause

Cloudera Director validation checks to ensure that special characters in configurations are enclosed in double quotes.

Solution

Use double quotes for special characters in configurations. An example of a configuration that would require double quotes is "log.dirs" in Kafka.

DNS Issues

Symptom

Director fails to bootstrap a cluster with a DNS error.

Cause

This can be caused by a couple of things:

- The **Edit DNS Hostnames** is not set to **Yes** the VPC settings.
- 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.

Solutions

In the AWS Management Console, go to **Services > Networking** and click **VPC**. In the VPC Dashboard, select your VPC and click **Action**. In the shortcut menu, click **Edit DNS Hostnames** and click **Yes**. If this does not fix the issue, continue with the instructions that follow to configure forward and reverse hostname resolution.

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 Does Not Start

Symptom

The Cloudera Director server does not 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 [Resource Requirements](#) on page 21 for more details on Cloudera Director hardware requirements.

Problem When Removing Hosts from a Cluster

Symptom

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

Cause

You are trying to shrink the cluster below the HDFS replication factor. See [Removing Instances from a Cluster](#) on page 131 (Note paragraph) 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, see [Setting up the AWS Environment](#) on page 24. For commands to turn off iptables, see either [Installing Cloudera Director Server and Client on the EC2 Instance](#) on page 27 or [Installing Cloudera Director Server and Client on Google Compute Engine](#) on page 40. 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 78.

How can I make Cloudera Director highly available?

Cloudera Director can set up highly available clusters in a Cloudera Manager deployment, but does not support a high availability setup for itself. You can make Cloudera Director more robust by configuring it to use a backed-up, robust MySQL database server (one that is hosted, for example, on AWS RDS) for its database instead of Cloudera Director's default H2 database. Then, if the Director instance goes down, another instance can be spun up that references the same database. In this case, Cloudera Director has the ability to resume interrupted work.

For information on setting up highly available clusters in a Cloudera Manager deployment using Cloudera Director, see [Creating Highly Available Clusters With Cloudera Director](#) on page 120.

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. Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure are cloud providers.

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.