cloudera®

Cloudera Director User Guide

Important Notice
© 2010-2018 Cloudera, Inc. All rights reserved.

Cloudera, the Cloudera logo, 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. If this documentation includes code, including but not limited to, code examples, Cloudera makes this available to you under the terms of the Apache License, Version 2.0, including any required notices. A copy of the Apache License Version 2.0, including any notices, is included herein. A copy of the Apache License Version 2.0 can also be found here: https://opensource.org/licenses/Apache-2.0

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.
395 Page Mill Road
Palo Alto, CA 94306
info@cloudera.com
US: 1-888-789-1488
Intl: 1-650-362-0488
www.cloudera.com

Release Information

Version: Cloudera Director 2.5.x
Date: June 29, 2018

Table of Contents

Introduction....10

Cloudera Director Features....10

Cloudera Director Interfaces....11

Web User Interface....11

Command-line interface....11

API....13

Stand-alone Client....13

Cloudera Director Interface Usage....14

Displaying Cloudera Director Documentation....15

Cloudera Director Release Notes....16

New Features and Changes in Cloudera Director....16

New Features and Changes in Cloudera Director 2....16

New Features and Changes in Cloudera Director 1....18

Known Issues and Workarounds in Cloudera Director....19

Usernames should support user@domain.com format....19

Tag on creation not available for AWS China and GovCloud....19

Cloudera Director may orphan AWS Spot instances....19

Added instance groups are not configured based on their instance type....19

Maximum length of environment name is misleading....19

Timeouts sending usage bundles....20

AWS rate limiting due to large number of EBS volumes....20

Cloudera Director cannot deploy Cloudera Navigator Key Trustee Server....20

Resize script cannot resize XFS partitions....20

Cloudera Director does not set up external databases for Sqoop2....20

Metrics not displayed for clusters deployed in Cloudera Manager 5.4 and earlier clusters....20

Changes to Cloudera Manager username and password must also be made in Cloudera Director....20

Cloudera Director may use AWS credentials from instance of Cloudera Director server....21

Root partition resize fails on CentOS 6.5 (HVM)....21

When using RDS and MySQL, Hive Metastore canary may fail in Cloudera Manager....21

Issues Fixed in Cloudera Director....21

Issues Fixed in Cloudera Director 2.5.1....21

Issues Fixed in Cloudera Director 2.5.0....22

Issues Fixed in Cloudera Director 2.4.1....24

Issues Fixed in Cloudera Director 2.4.0....27

Issues Fixed in Cloudera Director 2.3.0....28

Issues Fixed in Cloudera Director 2.2.0....31

Issues Fixed in Cloudera Director 2.1.1 32
Issues Fixed in Cloudera Director 2.1.0 33
Issues Fixed in Cloudera Director 2.0.0 34
Issues Fixed in Cloudera Director 1.5.2 34
Issues Fixed in Cloudera Director 1.5.1 34
Issues Fixed in Cloudera Director 1.5.0 34
Issues Fixed in Cloudera Director 1.1.3 35
Issues Fixed in Cloudera Director 1.1.2 35
Issues Fixed in Cloudera Director 1.1.1 35

Requirements and Supported Versions 37

Cloud Providers 37
Cloudera Director Service Provider Interface (SPI) 37
Supported Software and Distributions 37
Resource Requirements 38
Supported Cloudera Manager and CDH Versions 39
Networking and Security Requirements 39
Supported Browsers 40

Getting Started with Cloudera Director 41

Getting Started on Amazon Web Services (AWS) 41
Setting up the AWS Environment 41
Launching an EC2 Instance for Cloudera Director 42
Installing Cloudera Director Server and Client on the EC2 Instance 45
Configuring a SOCKS Proxy for Amazon EC2 48
Deploying Cloudera Manager and CDH on AWS 50
Pausing a Cluster in AWS 57
Cleaning Up Your AWS Deployment 58
Getting Started on Google Cloud Platform 58
Creating a Google Cloud Platform Project 58
Configuring Tools for Your Google Cloud Platform Account 58
Creating a Google Compute Engine VM Instance 60
Installing Cloudera Director Server and Client on Google Compute Engine 61
Configuring a SOCKS Proxy for Google Compute Engine 64
Deploying Cloudera Manager and CDH on Google Compute Engine 64
Cleaning Up Your Google Cloud Platform Deployment 69
Getting Started on Microsoft Azure 69
Obtaining Credentials for Cloudera Director 69
Setting up Azure Resources 70
Setting Up Dynamic DNS on Azure 75
Setting Up MySQL or PostgreSQL 82
Setting Up a Virtual Machine for Cloudera Director Server 83

Installing Cloudera Director Server and Client on Azure 83
Configuring a SOCKS Proxy for Microsoft Azure 85
Allowing Access to VM Images 86
Creating a Cluster 86
Adding New VM Images, Regions, and Instances 92
Important Notes About Cloudera Director and Azure 93

Usage-Based Billing 95

Prerequisites 95
How Usage-Based Billing Works 95
Deploying Cloudera Manager and CDH with Usage-Based Billing 96
Enabling Usage-Based Billing with the Cloudera Director Server web UI 96
Enabling Usage-Based Billing with bootstrap-remote 96
Managing Billing IDs with an Existing Deployment 97
Troubleshooting Network Connectivity for Usage-Based Billing 97
Cloudera Director Usage Bundles 98
Metadata Section 98
Cloudera Manager Block 98
Cloudera Director Block 99
Usage Logging 99

Customization and Advanced Configuration 101

The Cloudera Director Configuration File 101
Location of Sample Configuration Files 101
Customizing the Configuration File 101
Valid Role Types for Use in Configuration Files 101
Using the API to Import a Configuration File 101
Using Spot Instances 102
Planning for Spot Instances 102
Specifying Spot Instances 103
Spot Blocks: Specifying a Duration for Spot Instances 103
Best Practices for Using Spot Instances 104
Auto-Repair for Failed or Terminated Instances 104
Using Custom Repositories with Cloudera Manager and CDH 107
Creating a Custom Repository 107
Creating a Cloudera Manager and CDH AMI 107
Choosing an AMI 107
Finding Available AMIs 108
Running Cloudera Director and Cloudera Manager in Different Regions or Clouds 108
Using a New AWS Region in Cloudera Director 109
Entering the Region Code 109
Region Endpoints 109

Other Considerations 110
Using Products outside CDH with Cloudera Director 111
Custom Service Descriptors 111
Using Kudu with Cloudera Director 112
Using Spark 2 with Cloudera Director 113
Using Third-Party Products with Cloudera Director 114
Deploying a Java 8 Cluster 115
javaInstallationStrategy configuration 115
Bootstrap script 116
Creating AWS Identity and Access Management (IAM) Policies 116
Using MySQL for Cloudera Director Server 118
Installing the MySQL Server 119
Configuring and Starting the MySQL Server 119
Installing the MySQL JDBC Driver 121
Creating a Database for Cloudera Director Server 122
Configuring Cloudera Director Server to use the MySQL Database 122
Migrating Cloudera Director Server from H2 to MySQL 123
Step 1: Preparation of databases 123
Step 2: Export data from H2 database 123
Step 3: Prepare MySQL database for data import 130
Step 4: Import data to MySQL 131
Using MariaDB for Cloudera Director Server 131
Installing the MariaDB Server 131
Configuring and Starting the MariaDB Server 131
Installing the MariaDB JDBC Driver 133
Creating a Database for Cloudera Director Server 134
Configuring Cloudera Director Server to use the MariaDB Database 134
Cloudera Director Database Encryption 135
Cipher Configuration 135
Starting with Encryption 136
Changing Encryption 137
Using Amazon S3 Object Storage 138
Configuring Amazon S3 with Cloudera Director 138
Using EBS Volumes for Cloudera Manager and CDH 141
EBS Volume Types 141
Configuring EBS Volumes 142
Using an External Database for Cloudera Manager and CDH 143
Defining External Database Servers 144
Defining External Databases 150
Setting Cloudera Director Properties 153
Pausing Cloudera Director Instances 154
Pausing the Cloudera Director Server 154
Pausing a Cluster in Azure 154
Setting Cloudera Manager Configurations 155

Cluster Configuration Using Cloudera Manager *156*
Setting up a Cloudera Manager License *156*
Deployment Template Configuration *157*
Cluster Template Service-wide Configuration *158*
Cluster Template Roletype Configurations *158*
Configuring Cloudera Director for a New AWS Instance Type 159
Updated Virtualization Mappings *159*
Updated Ephemeral Device Mappings *160*
Using the New Mappings *160*
Configuring Cloudera Director to Use Custom Tag Names on AWS 160
Using the New Mappings *161*
Post-Creation Scripts 161
Deployment Post-creation Scripts *161*
Cluster Post-creation Scripts *162*
Predefined Environment Variables *164*
Enabling TLS with Cloudera Director 165
Creating Kerberized Clusters With Cloudera Director 165
Creating a Kerberized Cluster with the Cloudera Director Configuration File *166*
Creating Highly Available Clusters With Cloudera Director 167
Limitations and Restrictions *167*
Editing the Configuration File to Launch a Highly Available Cluster *167*
Using Role Migration to Repair HDFS Master Role Instances *169*
Enabling Sentry Service Authorization 170
Prerequisites *171*
Setting Up the Sentry Service Using the Cloudera Director CLI *171*
Setting up the Sentry Service Using the Cloudera Director API *172*
Related Links *172*

Managing Cloudera Manager Instances with Cloudera Director Server 173

Cloudera Director and Cloudera Manager Usage 173
When to Use Cloudera Director *173*
When to Use Cloudera Manager *173*
CDH Cluster Management Tasks *174*
Ensuring Consistency of Virtual Instance Groups *176*
CDH Cluster Management Guidelines for Cloudera Director *177*
Submitting a Cluster Configuration File 177
Ports Used by Cloudera Director 178
SSH Keys in Cloudera Director 179
The Role of Keys in SSH *179*
Use of SSH Keys in Cloud Providers *179*
Cloudera Director's Use of SSH Keys *180*
Good Practices for SSH Key Management with Cloudera Director *180*
Deploying Clusters in an Existing Environment 180

Cloudera Manager Health Information....181
Opening Cloudera Manager....182
Creating and Modifying Clusters with the Cloudera Director web UI....182
Configuring Instance Groups During Cluster Creation....182
Modifying the Number of Instances in an Existing Cluster....183
Repairing Worker and Gateway Instances in a Cluster....185
Terminating a Cluster....185
Terminating a Cluster with the web UI....186
Terminating a Cluster with the CLI....186
Diagnostic Data Collection....186
Manual Collection of Diagnostic Data....186
Configuring Diagnostic Data Collection....188
User Management....189
Managing Users with the Cloudera Director Web web UI....189
Managing Users with the Cloudera Director API....190

Cloudera Director Client....192
Installing Cloudera Director Client....192
Provisioning a Cluster on AWS....193
Running Cloudera Director Client....194
Connecting to Cloudera Manager with Cloudera Director Client....195
Modifying a Cluster with the Configuration File....196
Growing or Shrinking a Cluster with the Configuration File....196

Upgrading Cloudera Director....198
Before Upgrading Cloudera Director....198
Upgrading Cloudera Director....199

Troubleshooting Cloudera Director....203
Slow or Failed OS Updates in Some AWS Regions....204
Cloudera Manager API Call Fails....204
Cloudera Director Cannot Manage a Cluster That Was Kerberized Through Cloudera Manager....205
RDS Name Conflicts....205
New Cluster Fails to Start Because of Missing Roles....205
Cloudera Director Server Will Not Start with Unsupported Java Version....206
Error Occurs if Tags Contain Unquoted Special Characters....206
DNS Issues....206
Server Does Not Start....207
Problem When Removing Hosts from a Cluster....208
Problems Connecting to Cloudera Director Server....208

Frequently Asked Questions.....209
General Questions.....209

Cloudera Director Glossary.....211

Appendix: Apache License, Version 2.0.....213

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 transient clusters. With long running clusters, you deploy one or more clusters that you can scale up or down to adjust to demand. With transient 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.

For more information about Cloudera Enterprise in the cloud, see the Cloud documentation page.

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: • 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: • 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: • 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:

Benefit	Features
	• Multi-cluster health dashboard • Instance tracking for account billing

Cloudera Director Interfaces

Cloudera Director provides different user interfaces for centralized deployment, configuration, and administration of Cloudera Manager and CDH clusters in the cloud. After you complete the Cloudera Director installation, you can use any interface to deploy Cloudera Manager and CDH clusters in the cloud. To manage the CDH deployment, use the interface that is appropriate for the complexity of the configuration or administrative tasks that you need to perform.

Cloudera Director provides the following user interfaces:

- **Web User Interface (Web UI)** - The web UI is a graphical interface to deploy and manage clusters in the cloud. You can use the web UI to monitor the clusters and access the cluster activity logs.
- **Command-Line Interface (CLI)** - The command-line interface uses a configuration file to define the settings for a cluster. The configuration file allows you to deploy clusters with custom settings and without operator intervention.
- **API** - You can use the Cloudera Director API to programmatically control the lifecycle of your clusters. Cloudera Director provides SDKs for the Python and Java programming languages.
- **Standalone Client** - If you install the client without installing the server, you can use the client as a standalone application to deploy and manage clusters. The standalone client provides a command-line interface for deploying simple clusters that you can manage using the local commands. Use the standalone client to deploy CDH clusters for testing or development, not for production.

Web User Interface

After you install the Cloudera Director server, you can use a browser to access the Cloudera Director web UI.

The web UI has a dashboard that shows the available environments and displays information about the Cloudera Manager deployments and the clusters in the deployment. Use the setup wizard in the web UI to easily and quickly deploy clusters in the cloud. You can also use the web UI to define environments, deployments, and clusters, add nodes to clusters, or clone clusters.

When you use the web UI to deploy a cluster, Cloudera Director saves the state of the cluster in the Cloudera Director database. The database can store deployment information about multiple environments, deployments, and clusters that are deployed and managed by Cloudera Director. The deployment information in the database allows Cloudera Director to create additional clusters in the managed deployments.

By default, Cloudera Director saves deployment information in an H2 database. You can configure Cloudera Director to use an externally managed MySQL database. Specify an external database in the `application.properties` file in the server host.

If you use the web UI to deploy Cloudera Manager and CDH, you can use the web UI or API to manage the Cloudera Manager deployment, terminate clusters, or deploy additional clusters. You can use the command-line interface to deploy more clusters or to terminate clusters. You can also use the web UI to manage clusters if you use the command-line interface or API to deploy them.

You can use the web UI to perform any configuration or administrative task on a Cloudera Manager deployment. However, when you perform a complex or customized deployment or configuration, you might find it easier and more reliable to use the command-line interface with a configuration file or to use the API.

Command-line interface

To use the Cloudera Director command-line interface, you must install the Cloudera Director client in addition to the server. You can install the Cloudera Director client separately from the server. You can install the client in multiple locations, with all clients communicating with the same Cloudera Director server in the cloud.

When you run a command, the client connects to the server to complete the operation. To connect to the server, the client requires the host and user account information for the Cloudera Director server.

When you use the command-line interface to deploy a cluster, the state of the cluster is saved in the Cloudera Director database. The database can store deployment information about multiple environments, deployments, and clusters that are deployed and managed by Cloudera Director. The deployment information in the database allows Cloudera Director to create additional clusters within the managed deployments.

If you use the command-line interface to deploy a cluster, you can use the web UI or the API to manage the cluster.

Application Properties File

When you install the Cloudera Director client, the installation creates a configuration file named application.properties. The properties file includes configuration properties such as the Cloudera Director server host, port number, and user account. You can modify the settings in the application.properties file based on your operational requirements.

By default, the command-line interface reads the settings in the application.properties file on the client host to determine the parameters of a command. When you run a command, you can override properties in the application.properties file by passing the properties directly to the command. For example, you can pass the hostname and port number for the Cloudera Director server. If you do not include these properties in the command, the command reads the properties from the application.properties file.

Cluster Configuration File

A template is a common and useful way to define the configuration and infrastructure of a cloud deployment. Cloudera Director uses a configuration file as a template for cluster deployments in the cloud. You can use the configuration file to define your cluster deployment across different cloud environments.

The Cloudera Director command-line interface uses a configuration file to determine the deployment configuration for a cluster. When you use the command-line interface to deploy a cluster, you must provide the configuration file name. The command reads the file you specify and deploys a cluster configured with the settings defined in the configuration file.

You can create multiple configuration files to deploy clusters with different settings, or you can reuse a configuration file to deploy multiple clusters with the same settings. Cloudera provides sample configuration files that you can use as templates to start a configuration file for your cluster deployment. You can find the sample configuration files on the Cloudera Director scripts GitHub page.

Commands

The command-line interface includes the following commands:

Command	Description
`bootstrap-remote`	Creates an environment, deployment, and cluster on a remote server based on the settings in a configuration file. The configuration file name must have a .conf extension. The bootstrap-remote command reads the configuration file and creates a cluster with the configuration settings defined in the file. `bootstrap-remote` speeds up the bootstrap process by configuring Cloudera Manager and the CDH cluster in parallel. To ensure that the command connects to the Cloudera Director server correctly, you can pass server host and user account properties to the command. The bootstrap-remote command uses the values you pass to connect to the server instead of the values in the application.properties file. For example, you can pass the following properties: • `lp.remote.hostAndPort=host[:port]` Hostname and port number of the Cloudera Director server. The default value in the `application.properties` file is set to `localhost:7189` • `lp.remote.username=<Cloudera Director server username>`

Command	Description
	Username to use to log in to the Cloudera Director server. • `lp.remote.password=<Cloudera Director server password>` Password for the Cloudera Director server user account.
`terminate-remote`	Terminates a cluster and deployment on a remote server. As in the `bootstrap-remote` command, you can pass the hostname and port number to connect to the Cloudera Director server and the username and password to log in to Cloudera Director.
`validate`	Validates the configuration file of an environment, deployment, or cluster. For clusters,it validates the correctness of the role and service types, but not the configuration keys, values, or semantics of the role placement You can set the `lp.validate.verbose` property to `true` to output an HTML representation of the configuration.

API

Cloudera Director has an API that provides access to all Cloudera Director features. The Cloudera Director API is a REST API that uses JSON as the data interchange format.

Use the API to access Cloudera Director from a script or to integrate Cloudera Director features with an application. The API includes SDKs to help you integrate Cloudera Director into Python or Java applications. You can use the API to deploy Cloudera Manager and CDH clusters on any cloud environment supported by Cloudera Director. You can find information about the Cloudera Director Java and Python APIs on the Cloudera Director SDK GitHub page.

The API includes a console to assist the development process. You can use the API console during development to interactively configure settings or perform ad hoc operations on the cluster in the cloud. You can also use it to explore Cloudera Director features and to test and troubleshoot clusters. You can access the API console for your deployment at `http://director-server-hostname:7189/api-console`.

Stand-alone Client

You can install the Cloudera Director client without installing the server. If you install the Cloudera Director client only, you can run local client commands to deploy and manage simple clusters.

When you install the client, the installation creates a configuration file named application.properties locally. The client uses the settings in the local application.properties file to determine the parameters of the command. When you run a command, you can override properties in the local application.properties file by passing the properties directly to the command.

The stand-alone client also uses a configuration file to determine the deployment configuration for a cluster. You can define the configuration settings for a cluster setup in a configuration file and use the local bootstrap command to deploy clusters based on the settings in the configuration file. You can reuse a configuration file to deploy multiple clusters with the same configuration. Cloudera provides sample configuration files that you can use as templates to start a configuration file for your cluster deployment. You can find the sample configuration files on the Cloudera Director scripts GitHub page.

When you use a local command to deploy a cluster, the state of the environment and cluster is saved in a local H2 database. The database stores information only for a single environment and deployment and allows only a limited set of operations to be performed on the deployed cluster. All local commands operate on the cluster deployment that is described in the local database.

If you use the local bootstrap command to deploy a cluster, you cannot use the web UI or the API to manage the cluster. You must manage the cluster using the local client commands. Cloudera recommends that you use the local client commands to deploy and manage clusters for demonstrations and development, not for production.

You can use the following local commands to deploy and manage a simple cluster:

Command	Description
`bootstrap`	Creates an environment, deployment, and cluster based on the settings defined in a configuration file. The configuration file name must have a .conf extension. The `bootstrap` command is comparatively slower than `bootstrap-remote` command. This is because, while `bootstrap-remote` configures Cloudera Manager and the CDH cluster in parallel, `bootstrap` configures a CDH cluster only after it configures Cloudera Manager. **Note:** If you use the local bootstrap command to deploy a cluster, you cannot use the web UI or the API to update or manage the cluster.
`status`	Reports the status of an environment, deployment, and cluster.
`terminate`	Terminates a cluster or deployment. Requires a configuration file. You can set the `lp.remote.terminate.assumeYes` property. This property determines if the user must explicitly confirm termination (false) or if confirmation is assumed (true). The default value is false. Setting this property to true will cause termination to proceed even if diagnostic data collection has failed. For more information, see Diagnostic Data Collection on page 186
`update`	Updates an environment, deployment, and cluster. Requires a configuration file.
`validate`	Validates the configuration file of an environment, deployment, or cluster. For clusters,it validates the correctness of the role and service types, but not the configuration keys, values, or semantics of the role placement You can set the `lp.validate.verbose` property to `true` to output an HTML representation of the configuration.

Cloudera Director Interface Usage

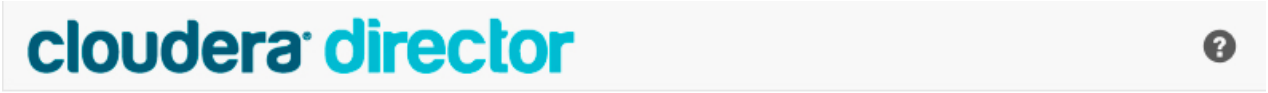
The following table shows the tasks you can perform in the different Director interfaces:

Task	Web UI	Command-line Interface	API	Stand-alone Client
Deploy simple clusters	■	■	■	■
Deploy complex clusters with Kerberos or high availability		■	■	
Deploy in production		■	■	
View dashboard of cluster deployment	■			
Manage multiple clusters	■	■	■	
Add nodes to clusters	■		■	■
Remove nodes from clusters	■		■	■
Clone clusters	■	■	■	

Task	Web UI	Command-line Interface	API	Stand-alone Client
Update Cloudera Manager password	■		■	■
Terminate clusters	■	■	■	■

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 at `/etc/cloudera-director-server/`.

If you edit the server `application.properties` file while Cloudera Director server is running, you must restart the server in order for your changes to take effect:

```
$ sudo service cloudera-director-server restart
```

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

What's New in Cloudera Director 2.5.1

- A number of issues have been fixed. See Issues Fixed in Cloudera Director 2.5.1 for details.

What's New in Cloudera Director 2.5.0

- Automatic repair: Cloudera Director can now automatically repair instances that get terminated unexpectedly for clusters that have auto-repair enabled.
- Significant improvement in resilience to unexpected instance terminations during bootstrap and grow operations. This improves support for AWS spot instances, as well as resilience of clusters using on-demand instances.
- Cloudera Director can now configure and create external accounts in Cloudera Manager that include cloud provider login credentials in order to manage the use of additional cloud provider functionality, including object stores, such as Amazon S3 in AWS.
- Cloudera Director now supports S3Guard, which guarantees a consistent view of data stored in Amazon S3.
- Cloudera Director can now recognize changes made directly from the cloud provider console to provisioned instances such as instance type and AMI.
- Cluster details page in the web UI now shows cluster history, displaying the most recent updates, including instances added, removed, or replaced in the cluster.
- Logging improvements, including additional information about service and role configuration errors returned by the Cloudera Manager API.
- A number of issues have been fixed. See Issues Fixed in Cloudera Director 2.5.0 for details.

What's New in Cloudera Director 2.4.1

- A number of issues have been fixed. See Issues Fixed in Cloudera Director 2.4.1 for details.
- Support for the following Microsoft Azure instance types:
  - STANDARD_D12_v2
  - STANDARD_D13_v2
  - STANDARD_D14_v2
  - STANDARD_D15_v2

  Refer to the Cloudera Reference Architecture on Azure for supported configurations using these Storage Account types.

What's New in Cloudera Director 2.4.0

- A number of issues have been fixed. See Issues Fixed in Cloudera Director 2.4.0 for details.
- Improved support for long-running clusters: Cloudera Director 2.4 will support upgrades, change of services, and configurations from Cloudera Manager 5.11.
- Support for Spark 2 and Kudu.
- Support for multiple bootstrap scripts. This feature allows you to combine functionality required during bootstrap while maintaining modularity. Using multiple scripts is beneficial when different parts of the customized bootstrap have different retry semantics, because retry upon failure is done only for failed scripts.

- Configurable AWS instance tag names and values.
- Support for passing of AWS user data to instances while they are being launched.
- Sample code using the Java SDK to grow, shrink and repair the cluster. See Cloudera Director Java Client Samples.
- For Azure clusters, support for Azure Data Lake Store via Cloudera Manager and CDH 5.11 deployments.
- Support for the following Microsoft Azure Clouds and Regions:
  - Azure Government: US GOV VIRGINIA
  - Azure Germany: Germany Central and Germany Northeast
- Added RHEL 6.8 to the default OS list for Azure.
- Improved error logging for Azure clusters.

What's New in Cloudera Director 2.3.0

- A number of issues have been fixed. See Issues Fixed in Cloudera Director 2.3.0 for details.
- Support for AWS Spot Blocks. Spot blocks can be used to ensure the duration of instances during bootstraps so that the bootstrap process is resilient to fluctuations in AWS Spot instance availability. For more information, see Specifying a Duration for Your Spot Instances in the Amazon AWS documentation.
- More control of cluster configuration with the introduction of instance-level and deployment-level post-creation scripts, in addition to existing cluster-level post-creation scripts.
- Updated look and feel of the Cloudera Director web user interface.
- Support for Cloudera Enterprise 5.10 out of the box.
- Support for RHEL 6.8 and RHEL 7.3 for Cloudera Director and CDH cluster instances.
- Support for the following Microsoft Azure instance types:
  - Standard_DS15_v2
  - Standard_GS4
  - Standard_GS5

  Refer to the Cloudera Reference Architecture on Azure for supported configurations using these Storage Account types.
- On Microsoft Azure, support for the CentOS 6.8 image published by Cloudera.

What's New in Cloudera Director 2.2.0

- A number of issues have been fixed. See Issues Fixed in Cloudera Director 2.2.0 for details.
- Support for AWS EBS volumes. See Using EBS Volumes for Cloudera Manager and CDH.
- On-demand and automatically-upon-failure diagnostic data collection. See Diagnostic Data Collection.
- Improved readability and validation of input configuration files.
- Support for Cloudera Enterprise 5.9 out of the box.
- Support for the following Microsoft Azure Storage Account types. Refer to the Cloudera Reference Architecture on Azure for supported configurations using these Storage Account types:
  - PremiumLRS
  - StandardLRS
- Support for Azure's P20 (512 GiB) disk type for Premium Storage Accounts. This is in addition to the already-supported P30 disk type. Refer to the Cloudera Reference Architecture on Azure for supported disk size configurations.
- On Microsoft Azure, support for the RHEL 7.2 image published by Red Hat in partnership with Microsoft and the CentOS 7.2 image published by Cloudera.

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 95.
- Running CM and CDH clusters on Microsoft Azure is supported. See Getting Started on Microsoft Azure on page 69
- 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 108.
- Cloudera Enterprise 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 virtual machine 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.
- 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 34 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 34 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 35 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.

Usernames should support user@domain.com format

When "." is used in a username, the Cloudera Director UI fails to create the user; Cloudera Director client does create a user, but the `get user` and `update user` commands return 404.

Tag on creation not available for AWS China and GovCloud

The AWS plugin in Cloudera Director 2.5 now supports tagging instances on creation for reliability purposes. However, neither the AWS China region nor AWS GovCloud currently support this functionality. For more information about tagging on creation for AWS instances, see the AWS blog article Tag EC2 Instances & EBS Volumes on Creation.

Workaround: Use Cloudera Director 2.4 if you require tag-on-create functionality for these regions until the regions have been updated by AWS.

Cloudera Director may orphan AWS Spot instances

Cloudera Director tags AWS instances with an instance id. Due to eventual consistency in the AWS API, Cloudera Director may not find the tagged instances during the bootstrap or update process.

Added instance groups are not configured based on their instance type

Newly added instance groups are not automatically configured. Roles on the new instances will be given the same configuration as existing roles even if the new instance uses a different instance type than the old instances.

Workaround: Update the role group in Cloudera Manager after the cluster update completes.

Maximum length of environment name is misleading

Parsing of a Cloudera Director configuration file can produce errors indicating that the environment name and, sometimes additionally, the deployment name are too long, even though the names were not specified in the file. This is a consequence of Cloudera Director automatically generating those names, based on the cluster name, when they

are not explicitly given in the file. When the cluster name is long, the automatically-generated names exceed the length limits.

Workaround: Provide explicit names for the environment and deployment in the configuration file. Examples:

```
environmentName: MyEnvironment
deploymentName: MyDeployment
```

Timeouts sending usage bundles

When usage-based billing is enabled, Cloudera Director may eventually cease sending usage bundles. An error similar to this is shown in the Director log:

```
WARN [InternalReportingService RUNNING] -
c.c.m.r.u.g.t.TapeUsageBundleReportingService: Failed to
dispatch bundle UsageBundleMetadata{version='v1',
licenseKey='REDACTED', ...}.
com.cloudera.metering.reporting.usage.generation.dispatch.
UsageBundleDispatchException: New upload request failed
...
Caused by: java.util.concurrent.TimeoutException: Timeout waiting for task.
```

Workaround: Restart Cloudera Director. Any usage bundles that are queued for submission before restart will be sent, and normal usage collection should resume.

AWS rate limiting due to large number of EBS volumes

Standing up a cluster with a large number of EBS volumes might trigger rate limiting on EBS allocation requests. The effect can spread to other calls from Cloudera Director to AWS.

Workaround: No more than 10 EBS volumes should be attached at a time.

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: Set up Cloudera Navigator Key Trustee Server via Cloudera Manager if using Cloudera Director 2.4 and above.

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. For more information on keeping Cloudera Director and Cloudera Manager in sync, see CDH Cluster Management Tasks.

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

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.

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 2.5.1

Custom tag mapping renames user-supplied tag names

The custom tag mapping functionality in the AWS plugin inadvertently renames user-supplied tags as well. For example: if a custom tag mapping is given to rename the "Name" field to "Director_Name," a user-defined tag of "Name" will also be renamed to "Director_Name."

Workaround: Utilize `instanceNamePrefixes`, which can simulate a custom name, albeit with a UUID at the end of it.

iptables modules blacklisted during normalization

The `iptables.ko` and `iptables_filter.ko` modules are blacklisted after instance normalization. This can conflict with applications like Docker, which utilize iptables for their own networking.

Workaround: Delete the `/etc/modprobe.d/iptables-blacklist.conf` file on each instance using an instance post create script.

Unable to grow cluster when upgrading Cloudera Manager configured with custom repository URLs

If Cloudera Manager in Cloudera Director was upgraded using a non-archive.cloudera.com URL and the cluster is grown, agent installation can fail on the new instances.

Failure of AWS Spot instance allocation may cause bootstrap or update to fail

Cloudera Director deletes Spot instance requests after attempting to allocate Spot instances. A Spot instance request may fail due to AWS API eventual consistency, causing the overall bootstrap or grow process to fail.

Incorrect retrieval of SSH credentials

If the SSH username is overridden in the instance template it can lead to an SSH authentication exception.

The setting preemptiveBasicProxyAuth did not work as expected

The `lp.proxy.http.preemptiveBasicProxyAuth` setting had no effect before.

Issues Fixed in Cloudera Director 2.5.0

AWS IAM permission for RDS required even when RDS not in use

When Cloudera Director validates an environment definition, it performs a call to AWS that requires the `rds:DescribeDBSecurityGroups` IAM permission. This is true whether or not RDS is to be used for any deployments or clusters in the environment.

Workaround: Include the `rds:DescribeDBSecurityGroups` permission in the IAM permissions for the user account defined in the environment; if no user is defined, then include the permission in the permission for the IAM role associated with the instance profile of the Director instance.

Cloudera Director masks real reason for cluster bootstrap or update failure during host installation

When Cloudera Director tries to add a host to Cloudera Manager, and errors occur that trigger retry, Cloudera Director can produce an exception that includes the following message while retrying:

```
'JdbcSQLException: Value too long for column "CALL_STACK'
```

This is an internal error that masks the root cause, which may be found earlier in the Cloudera Director log, or in the Cloudera Manager log.

Preemptive proxy authentication not working as expected

Preemptive proxy authentication does not work as intended and currently has no effect.

Workaround: Use the proxy without preemptive proxy authentication.

Restrictive umask prevents bootstrap

Cloudera Director inappropriately sets the file ownership when creating database scripts for use by Cloudera Manager. When using a more restrictive umask with Cloudera Director, this can prevent deployment bootstrap from succeeding properly.

Workaround: Use the default umask supplied with your filesystem, or use a more permissive umask.

First run fails for cluster with HA enabled for HDFS cloned through Cloudera Director

If you create a non-HA cluster through Cloudera Director with a Cloudera Manager version prior to 5.12, and then enable HA using the Cloudera Manager wizard, Cloudera Director cannot clone the cluster.

Workaround: In order to be able to clone such a cluster, update Cloudera Manager to 5.12 or later, and wait for Cloudera Director to refresh its internal cluster model, before enabling HA through the Cloudera Manager wizard.

Presence of external databases prevents deletion of environments

When you delete a database server, we check to see if it is an external db or a registered db. If it is external, we show the delete action: 'Terminate Database Server.' The delete action for an external database should be shown as "Unregister Database Server." Once the external database is unregistered, the environment can be safely deleted.

Refreshing a deployment may overwrite the deployment template

User updates to the deployment template for a deployment may be inadvertently overwritten by refreshing the deployment if they are performed at the same time. This currently only impacts Cloudera Manager credentials.

Workaround: Re-update the Cloudera Manager credentials.

Cloudera Director may fail to repair master roles for certain services when repairing an instance

When repairing an instance with master roles, Cloudera Director may incorrectly attempt to automatically configure those roles instead of running the role migration. This may lead to update failure.

Use numberOfHealthCacheExecutorThreads when provided

Previous versions of Cloudera Director ignored `numberOfHealthCacheExecutorThreads` when specified in the server configuration file.

Use numberOfCacheExecutorThreads when provided

Previous versions of Cloudera Director ignored `numberOfCacheExecutorThreads` when specified in the server configuration file.

Terminating cluster creation at the wrong time can leak instances

EC2 instances can potentially leak instances when terminated during deployment or cluster bootstrapping.

Workaround: Delete the leaked instances manually.

Unhealthy host(s) causes 'apply host template' failure when growing the Cluster

When growing an existing cluster, the update operation may fail to add the instances. If the server log shows `API call to Cloudera Manager failed. Method=HostTemplatesResource.applyHostTemplate`, the user can check the CM API debug logs. One of the reasons for failure could be that the CDH parcel hasn't been activated by the time Cloudera Director attempts to apply the host template. This specific scenario is likely to happen if newly added instances show up as unhealthy in Cloudera Manager, which can cause parcel distribution errors.

Workaround: The best course of action is to try to figure out why the newly-added instance(s) comes up as unhealthy. This can sometimes be fixed by using a different AMI or instance type. If that doesn't work, Cloudera Director's `lp.update.sleepTimeForAddInstanceSeconds` server property (added in Cloudera Director 2.4.1) can be increased to add additional time for the host to come back as healthy so that the parcel gets distributed and activated before the API call to `apply host template`.

Shrink operation fails to complete

Host decommissioning can hang in Cloudera Manager, causing a cluster shrink operation to fail to complete successfully.

Cluster termination fails during backup of Cloudera Manager configuration files

When a cluster is terminated while Cloudera Director is backing up Cloudera Manager configuration files, it is possible for Cloudera Director to hang attempting to clean up the associated pipelines.

Erroneous error message during cluster creation with Spark 2

When creating a cluster that includes the Spark 2 service with bootstrap-remote, the Cloudera Director client will display the following warning:

```
Found warnings in cluster configuration: Unknown role type:
GATEWAY for service type: SPARK2_ON_YARN in instance group
```

The warning is a false positive, but it does not stop the cluster creation.

Failure to update IP addresses in Cloudera Manager for repaired instances

When repairing instances, Cloudera Director will try to correlate instances known to Cloudera Director with instances known to Cloudera Manager. The correlation is done via the instance's IP address. However if the instance is terminated,

the IP address known to Cloudera Director will be a placeholder while Cloudera Manager keeps its original IP address, resulting in failure when attempting to establish the mapping.

Cloudera Director continues trying to update a cluster even if the cluster was terminated in the middle of the update

When updating a cluster, Cloudera Director will first check the cluster status, and compute update steps if it passes the check, and start pipelines to update the cluster with the update steps. However, the check and launching pipelines is not transactional, so if a terminate cluster request is fulfilled in the middle, the update will still kick off the pipeline.

Update fails if 'Redeploy Client Configuration' is checked unnecessarily

Update fails when the **Redeploy Client Configuration** checkbox on the **Modify Cluster** page is checked and redeployment of client configurations is not needed.

Workaround: Do not check the **Redeploy Client Configuration** checkbox if redeployment of the client configuration is not needed.

Cluster bootstrap fails when Spot instance capacity is exceeded

If the number of Spot instances requested exceeds the user's Spot capacity, then the allocation of the Spot instances will fail and cause cluster bootstrap to fail.

Workaround: Ensure your Spot instance limit on your EC2 account is sufficient for the number of instances you request in the specified region.

Cloudera Director may leak instances in AWS

Cloudera Director may retry a failed instance allocation, resulting in two instances tagged with the same ID. Due to the tagging, Cloudera Director may terminate only one of the instances.

Cloudera Director client does not support unicode

HOCON substitution in Cloudera Director configurations is not supported.

Workaround: Write configurations without substitutions.

Block volume limit error reporting

If the EBS volume limit is reached when creating a cluster, the Cloudera Director log might not reflect this root cause, though it might mention creating the cluster failed because it cannot satisfy the minimum threshold limit for specific roles in the cluster.

Issues Fixed in Cloudera Director 2.4.1

Errors when using MySQL 5.7 for the Cloudera Director database

The defaults related to `TIMESTAMP` field handling changed drastically in MySQL 5.7.4 and later, which is documented in SQL Mode Changes in MySQL 5.7 in the MySQL documentation. One of the tables created by Cloudera Director, `SERVER_CONFIGS`, conflicts with these new defaults, which were valid in previous versions of MySQL. This is further complicated by the fact that MySQL 5.7.x will allow upgrades from MySQL 5.6 with tables that violate these defaults.

The result is that any modifications attempted on the SERVER_CONFIGS table in MySQL 5.7.x will fail. Cloudera Director 2.4 introduced a change to this table, triggering this problem. Additionally, new installs have been observed failing on MySQL 5.7.x due to the `SERVER_CONFIGS` table violating the expected defaults.

This issue has been fixed in Cloudera Director 2.4.1 with database changes that:

- Adjust the creation of the `SERVER_CONFIGS` table on new installations
- Correct `SERVER_CONFIGS` for users upgrading to Cloudera Director 2.4
- Correct `SERVER_CONFIGS` for users who have already upgraded to Cloudera Director 2.4

For fresh installations on MySQL 5.7.x, this may affect any version of Cloudera Director starting with version 2.0. For existing installations that are now running on MySQL 5.7.x, this may affect users attempting to upgrade to Cloudera

Director 2.4 from Cloudera Director versions 2.0 to 2.3. Running on MySQL 5.5.x or 5.6.x will behave as expected without any database failures.

Workarounds:

- Cloudera recommends contacting Cloudera Support in order to fix this issue. However, if that is not an option, the following steps can be used to address the issue.
- For a fresh install of Cloudera Director, the simplest workaround is to disable strict mode on MySQL. For more information about strict mode and how to disable it, see SQL Server Modes in the MySQL documentation. Using Cloudera Director 2.4.1 will avoid this issue.
- For existing installs, manually modify the MySQL database to avoid this issue:
  - **Upgrading from versions lower than 2.0.0:** In this case, Cloudera Director will fail when trying to create the `SERVER_CONFIGS` table. In the database housing the Cloudera Director tables, examine the core_schema_version table and remove the line with the script value `V3_2.0.0_1__init_serverconfig.sql`.

```
delete from core_schema_version where script = 'V3_2.0.0_1__init_serverconfig.sql';
```

    You should see a response like the following:

```
Query OK, 1 row affected (0.02 sec)
```

    After this, retry the upgrade using Cloudera Director 2.4.1, or disable strict mode.
  - **Upgrading from versions 2.0.0 to 2.4.0:** In this case, Cloudera Director will fail when trying to modify several tables to remove the VERSION column. You must complete the migration manually and fix the TIMESTAMP issue.

```
ALTER TABLE SERVER_CONFIGS MODIFY UPDATED_AT TIMESTAMP NULL, MODIFY CREATED_AT TIMESTAMP
 NULL;

ALTER TABLE AUTHORITIES DROP COLUMN VERSION;
ALTER TABLE CLUSTERS DROP COLUMN VERSION;
ALTER TABLE DEPLOYMENTS DROP COLUMN VERSION;
ALTER TABLE ENVIRONMENTS DROP COLUMN VERSION;
ALTER TABLE EXTERNAL_DATABASE_SERVERS DROP COLUMN VERSION;
ALTER TABLE INSTANCE_TEMPLATES DROP COLUMN VERSION;
ALTER TABLE SERVER_CONFIGS DROP COLUMN VERSION;
ALTER TABLE USERS DROP COLUMN VERSION;

UPDATE core_schema_version set success = 1 where script =
'V3_2.4.0_1__remove_versions.sql'
```

    One or more of the ALTER TABLE statements may fail with an error that looks like the following:

```
ERROR 1091 (42000): Can't DROP 'VERSION'; check that column/key exists
```

    This can be ignored because it was correctly deleted as part of the initial attempt to upgrade to Cloudera Director 2.4.

    After this, retry the migration. Cloudera recommends upgrading to Cloudera Director 2.4.1 as soon as possible, although these manual corrections should alleviate the issue.

Bootstrap fails because of empty parcel list

Cloudera Director fails in the middle of bootstrap with `IllegalArgumentException: Parcel validation failed`. This can happen when Cloudera Manager instances take longer than usual to refresh the list of parcels.

Unhealthy host causes "apply host template" to fail when growing the cluster

When growing an existing cluster the update operation may fail to add the instances. If the server log indicates "API call to Cloudera Manager failed. Method=HostTemplatesResource.applyHostTemplate," the user should enable Cloudera

Manager API Debugging and check the server logs in Cloudera Manager to get more information on the failure. See Cloudera Manager API Call Fails in Troubleshooting Cloudera Director for information about checking Cloudera Manager logs.

One of the reasons for failure could be that the CDH parcel wasn't activated by the time Cloudera Director attempted to apply the host template. This specific scenario is likely to happen if newly added instances show up as unhealthy in Cloudera Manager.

Workaround: The best course of action is to try and figure out why the newly added instances comes up as unhealthy. This can sometimes be fixed by using a different AMI or instance type. If that doesn't work, Cloudera Director's `lp.update.sleepTimeForAddInstanceSeconds` server property can be increased to add additional time for the host to come back as healthy so that parcel gets distributed and activated before the API call to apply host template.

Azure VMs with manually attached Public IPs from different Resource Groups are marked as "not found"

An Azure VM with a manually attached Public IP from a different Resource Group will no longer be marked as "not found" and excluded from the the list of active cluster nodes. As of 2.4.1 the VMs will not report Public IPs from different Resource Groups but they will function as expected otherwise.

Workaround: Create the Public IP for manually attaching to the VM in the same Resource Group as the VM itself.

NullPointerException when Cloudera Director retrieves the private FQDN of a VM instance

In some rare cases the **OS Profile** metadata of an Azure VM can be empty. This can be confirmed by inspecting the VM metadata on Azure Resource Explorer ("osProfile" JSON block will be missing from the VM **properties** block). The OS Profile contains information such as the VM's private FQDN. An empty OS Profile can be related to Azure VM agents not running correctly on the VM. Cloudera recommends contacting Microsoft Azure support to resolve the issue where OS Profile is empty for an Azure VM. As of Cloudera Director 2.4.1, VM with missing OS Profile will no longer cause NullPointerException.

Add D series instances to Azure instance type list

The following instance types are added to to the Azure instance type list:

- STANDARD_D15_v2
- STANDARD_D14_v2
- STANDARD_D13_v2
- STANDARD_D12_v2

Expand Error Log for Unsupported Azure VMs

When deploying an unsupported Azure VM type the error message now contains actionable information for how to get and use the latest supported VM types.

Shorten Azure VMs Instance ID field in Cloudera Director UI

Azure VMs in Cloudera Director reported their instance IDs as a full Resource ID with Subscription ID and Resource Group name included. As of 2.4.1 the instance ID field is shortened to just the VM name.

Use static private IP address assignment option instead of dynamic

To guarantee the private IP address does not change after the VM is deallocated and restarted, the private IP allocation method must be **Static**. As of 2.4.1 the default private IP address allocation method is changed to **Static**.

Workaround: Manually change the private IP address assignment option to "Static" for each VM in the cluster via Azure portal.

Issues Fixed in Cloudera Director 2.4.0

Root password for external database server emitted in log

Cloudera Director logs the command line it runs to create new databases for Cloudera Manager and for cluster services. As of version 2.3, the password for the database being created was redacted in these log messages, but the password for the root account of the database server was not. This is fixed in 2.4, and the root password is now also redacted.

Cloudera Director may show the status of a cluster as TERMINATE_FAILED even when it has successfully terminated

If a cluster is terminated while in the process of bootstrapping, it is possible for the cluster to show **TERMINATE_FAILED** even though it has successfully terminated.

Cloudera Director does not sync with 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. For more information on keeping Cloudera Director and Cloudera Manager in sync, see CDH Cluster Management Tasks.

Old pipeline records not evicted from the Cloudera Director database

Cloudera Director records data about its internal workflow pipelines in its own database. Persisting this information allows Cloudera Director to track pipeline progress across restarts and to resume pipelines that were running or suspended. Pipeline data for old pipelines, such as those that have completed or failed, is automatically evicted from this database. However, under some circumstances, old pipeline data would fail to be evicted, resulting in logged errors. One cause is a Cloudera Director restart, which destroys in-memory pipeline data that was erroneously expected to remain. Cloudera Director 2.4 is more robust and eliminates this cause of pipeline eviction failure.

Workaround: In Cloudera Director 2.3 and below, the inability to evict old pipeline data does not harm Cloudera Director functioning in the short term, but over time the database could grow unacceptably large. Contact Cloudera Support for assistance deleting pipelines that cannot be evicted normally. To prevent build-up of old pipeline data, do not stop Cloudera Director until a round of database eviction is complete.

Delete deployment may orphan underlying clusters

When deleting deployments, it is possible that Cloudera Director deletes a deployment successfully, but leaves the cluster in an undeleted state. Retrying deployment deletion will not help, and the clusters will be orphaned. This is fixed in 2.4 such that a deployment deletion will also check for any orphaned clusters, even if the deployment itself is deleted.

Workaround: In Cloudera Director 2.3 and below, individually delete orphaned clusters if there ID's are known.

Bootstrap fails with non-default password-protected parcel repository

Bootstrap fails when using a password-protected CDH parcel repository with Cloudera Director 2.3 and below. This has been corrected in Cloudera Director 2.4.

Cloudera Director bootstrap hangs if EC2 spot instances terminate immediately after fulfillment

With Cloudera Director 2.3 and below, bootstrap can hang if spot instances terminate immediately after fulfillment, making it necessary to cancel the cluster bootstrap, terminate the cluster, and try again. This has been corrected in Cloudera Director 2.4 such that bootstrap fails immediately.

NullPointerException thrown when creating an invalid environment on Azure

In Cloudera Director 2.3.0 and below, a `NullPointerException` is thrown when invalid Microsoft Azure environment information (Subscription ID, Tenant ID, Client ID or Client Secret) is used in creating a new Azure Environment. For Cloudera Director 2.4.0 and higher, an error message is shown indicating that invalid Azure environment information was used to create the new Azure environment.

Terminated host not properly cleaned up during shrink or repair

When shrinking or repairing an instance that has been terminated outside of Cloudera Director, Cloudera Director may fail to decommission and delete the host from Cloudera Manager.

Terminated EC2 instances report 127.0.0.1 as private IP

AWS instances that were terminated outside of Cloudera Director may have reported an IP address of 127.0.0.1. This has been changed in Cloudera Director 2.4 so that the IP address 192.0.2.1 is reported (an IP address reserved for documentation).

Cloudera Director client infinitely tries to create services if you specify duplicate services

If duplicate services are specified for a cluster (for example, two Hive services or two Impala services), Cloudera Director will infinitely retry to create services during cluster creation.

Workaround: Cancel the cluster bootstrap, terminate the cluster, and recreate without duplicate services.

Cloudera Director may not apply custom configuration to all instances

Cloudera Director requests that Cloudera Manager perform automatic configuration for a cluster prior to applying any custom configurations. Automatic configuration may sometimes create multiple groups of instances within Cloudera Manager for a single corresponding group requested by Cloudera Director. When this occurs, custom configurations for the instances will only be applied to instances in one of the Cloudera Manager groups.

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.

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.

Cluster launch fails using the development version of Cloudera Manager 5.10 and CDH 5.10 with Kudu

Cloudera Director 2.3 does not support deployment and management of Kudu.

If a cluster is terminated while it is bootstrapping, the cluster must be terminated again to complete the termination process

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

Severity: Low

Issues Fixed in Cloudera Director 2.3.0

Deployment bootstrap process may fail to complete

The process of bootstrapping a deployment can hang indefinitely waiting for Cloudera Manager to start, even after Cloudera Manager is up and reachable.

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.

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.

Normalization process does not set swappiness correctly on RHEL 7.2

On CentOS/RHEL 7 operating systems, the tuned service overwrites the swappiness settings that Cloudera Director configures on instances.

Stale service configs

Cloudera Director sometimes fails to detect stale services properly when restarting a cluster.

The nscd tool is installed but not enabled during normalization

`nscd`, a tool which caches common name service requests, is installed on Cloudera Director-managed instances, but is not enabled on CentOS and RHEL. This can reduce the performance of the bootstrapping process.

Cluster update or termination during instance metadata refresh fails to complete

If a deployment or cluster is terminated or updated at the same time that a refresh of instance metadata is running, on rare occasions the refresh will prevent the terminate or update operation from completing properly.

Director detects SRIOV incorrectly

For AWS instances, Cloudera Director will always report Enhanced Networking (SR-IOV) as `false` (for example on the instance properties page), even when it's enabled. This is fixed in Cloudera Director 2.3 and requires IAM permissions for the EC2 method `DescribeInstanceAttribute`.

After Cloudera Manager bootstrap failure, termination leads to renewed bootstrap attempt

In Cloudera Director 2.2, if you attempt to terminate a cluster or deployment in the BOOTSTRAP_FAILED stage, it may go back into the BOOTSTRAPPING stage and return the following exception message: `java.util.concurrent.TimeoutException: Pipeline did not complete in 10 SECONDS`. In this situation, terminating the deployment or cluster a second time should terminate the cluster or deployment as expected. This can also happen in Cloudera Director 2.1, but the exception message will be the following more generic message: `500 internal server error`.

Warning when adding Hue Load Balancer role

When you bootstrap or validate a cluster that has the HUE_LOAD_BALANCER role, Cloudera Director generates an unknown role type warning for the role.

Bootstrap failure with Kafka and Sentry on Cloudera Manager 5.9

Cluster bootstrap fails when using Cloudera Manager 5.9 with both Kafka 2.0 and Sentry.

If Kafka and Sentry are required on the same cluster, use one of the following combinations:

- Kafka 2.1 with Cloudera Manager 5.9 or 5.10
- Kafka 2.0 with Cloudera Manager 5.8 or lower

Lack of support for newer AWS regions

When selecting certain AWS regions, such as `ap-northeast-2`, an error message can appear stating **Unable to find the region ap-northeast-2**. In this case, manually set the KMS region endpoint (under **Advanced Options**) to the KMS region endpoint specified in the AWS Regions and Endpoints in the AWS documentation.

Cloudera Manager repository URL validation failure

The validation of the Cloudera Manager repository can fail during the bootstrap process if the URL uses a host like `localhost`, a single-word hostname, or one with an internal or non-standard domain name. Use an IP address for the host, or use a hostname with a common domain like `.com`.

Cloudera Director configures Hue to use SQLite

CDH 5.8 and higher installs Postgres drivers along with Hue. When configuring a cluster to use Cloudera Manager's embedded Postgres database, Director will configure Hue to use its own embedded SQLite database rather than Cloudera Manager's embedded Postgres database.

MySQL database creation fails with insufficiently strong password

When using MySQL 5.7 as an external database server for a Cloudera Director deployment or cluster, database creation may fail with an error: "Your password does not satisfy the current policy requirements." This is due to Cloudera Director generating random UUIDs for passwords, which do not satisfy the MEDIUM level of password validation in MySQL 5.7. Disable password validation in MySQL, or adjust the validation level to LOW.

RDS instance creation fails with password length violation

AWS RDS requires a master user password of at least eight characters. If a password is supplied that is too short, Cloudera Director fails to validate it, leading to a failure from RDS. Ensure that the master user password is at least eight characters long.

Cloudera Manager server logs in Diagnostic data may be empty

Cloudera Director automatically attempts to collect diagnostic data after cluster bootstrap failure. If cluster bootstrap failed before or just after the cluster is created in Cloudera Manager, then the `scm-server-logs` inside the diagnostic data may be empty. In this case, trigger diagnostic data collection on the deployment.

High Azure Standard Storage Disk Usage

Azure Standard Disks are billed for used space + transactions (see Azure Storage Standard Disk Pricing). In Cloudera Director 2.2, Standard Storage Virtual Hard Disks (VHDs) are mounted without the "discard" option. As a result, if a file is deleted on the VHD it does not release this space back to Azure Standard Storage and it will continue to be billed as used space. Note: this issue does not cause disk space leakage; space occupied by deleted files can still be used by new files.

To address this problem, edit the `prepare_unmounted_volumes` file to add the `discard` mount option. `prepare_unmounted_volumes` is located at `/var/lib/cloudera-director-plugins/azure-provider-1.1.0/etc/`.

Change line 78 from:

```
echo "UUID=${blockid} $mount $FS defaults,noatime 0 0" >> /etc/fstab
```

to

```
echo "UUID=${blockid} $mount $FS defaults,noatime,discard 0 0" >> /etc/fstab
```

Restart the Cloudera Director server service after making this change.

Java Clients return null for a 404 ("Not Found") response

The Java client currently can return null values for both 204 and 404 response codes from the `collectDiagnosticData` service endpoint. Therefore, it is difficult to tell if a collection call fails because a deployment or cluster is missing. In this case, poll for the status for a finite amount of time. If the poll times out, consider the collection attempt failed.

Incorrect choice of response code for cluster update failure

An API request to update a cluster fails if the cluster is in transition, for example, if it is already being updated. The response code for the failure, however, is 204, which indicates success.

Environments may not be able to be deleted temporarily

Even when an environment is empty, that is, all of its deployments and external databases have been deleted, it can take five to ten minutes before it is possible to delete the environment. This is due to remaining data structures that have not yet been automatically cleaned up.

SELinux remains enabled on instances allocated by Director

Depending on the operating system, Cloudera Director may misread the state of SELinux on instances it allocates and determine that it is disabled, when it is actually still enabled. This can lead to errors running Cloudera Manager or cluster services.

Security group validation should be configurable

This change provides a new capability for end users to enforce network requirements. It allows users to define the network rules configuration and validates AWS security groups against the pre-defined rules. When writing rules, users can not only define allowed networking traffic, but also deny traffic against specific ports from a list of IP ranges.

Time daemons do not run properly on RHEL and CentOS 7.x instances

The choice of standard time daemon for RHEL and CentOS 7.x releases has changed from `ntpd` to `chronyd`. However, Cloudera Director does not perform the correct commands when normalizing instances to properly set up `chronyd`. Instances may end up with `ntpd` running, or no time daemon running at all. To avoid this, rely on `ntpd` for time synchronization, and use an instance bootstrap script to disable `chronyd` and enable `ntpd`. For more information, see Configuring NTP Using NTPD in the Red Hat Linux 7 System Administrator's Guide.

Issues Fixed in Cloudera Director 2.2.0

Storage Encryption for AWS RDS Instances

Before Cloudera Director 2.2, storage encryption for AWS RDS instances was not supported, despite the presence of a KMS key ID field in the web UI form for describing RDS instances. The web UI field was ignored. In Cloudera Director 2.2, storage encryption is supported, using the default key ID associated with RDS for the AWS account. Use of a non-default KMS key is not supported, and the KMS key ID field has been removed from the web UI. See Defining External Database Servers for information on enabling storage encryption for a new RDS instance.

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.

Azure operation timeout

Some Azure operations, such as VM creation and deletion, can take longer to complete than the default timeout value of 20 minutes. When this occurs, the Cloudera Director Azure plugin will timeout the Azure operation, resulting in a failure to complete the operation. Adjusting the Cloudera Director server timeout does not help.

Wait until Azure operation time drops back to normal range (less than 20 minutes).

Affected Versions: Cloudera Director 2.1.0, 2.1.1. Beginning in Cloudera Director 2.2.0, the user can change the timeout value for Azure if the default value of 20 minutes is not long enough.

Deployment fails on Azure due to incompatible instance type existing in an Availability Set

VM creation fails if the VM of one series (for example, DS13) is deployed into an Azure Availability Set that already contains one or more VMs from a different series (for example, DS13_V2). This is an Azure platform restriction.

Affected Versions: Cloudera Director 2.1.0, 2.1.1. Beginning in Cloudera Director 2.2.0, an error is reported when an instance template is created that will cause a VM to be deployed into an incompatible Availability Set.

Add check to make sure resources are in the same region

VM creation fails when using resources from one region (for example, a VNET in EastUS) to deploy a VM in another region (for example, WestUS). This is an invalid configuration yet it may not be obvious when configuring an instance template.

Affected Versions: Cloudera Director 2.1.0, 2.1.1. Beginning in Cloudera Director 2.2.0, an error will be shown if the user tries to configure an instance template with resources from a different region than what is defined at the environment level.

Some valid host FQDN suffixes are not allowed in the Azure instance template

The regex check for the host FQDN suffix (DNS domain on the private cluster network) does not allow valid host FQDN with fewer than three characters. For example, `company.us` is not allowed.

Affected Versions: Cloudera Director 2.1.0, 2.1.1. Beginning in Cloudera Director 2.2.0, the check for host FQDN has been relaxed to allow names like `company.us` or `company.1.us`.

Merge user-provided image configuration files with internal ones

Updating a Cloudera Director Azure plugin configuration file (`images.conf`) requires replacing the entire configuration file, even if only part of the configuration file needs to be updated.

Affected Versions: Cloudera Director 2.1.0, 2.1.1. Beginning in Cloudera Director 2.2.0, the user can provide partial Azure plugin configuration files containing only the portions to be updated.

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, 6.8, 7.1, 7.2, and 7.3 Ubuntu 14.04	**For AWS and Google Cloud Platform:** RHEL and CentOS 6.5, 6.7, 6.8, 7.1, 7.2, and 7.3 **Spot instances:** AWS supports Spot instances only on Centos operating systems, not on RHEL. **For Microsoft Azure:** RHEL and CentOS 6.7, 6.8, and 7.2 RHEL 7.2 is supported only for Cloudera Manager and CDH 5.7 and higher. RHEL 7.3 (on AWS) is supported only for Cloudera Manager and CDH 5.10 and higher. 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	Oracle JDK version 7 or 8

	Cloudera Director	Cloudera Manager and CDH
	For download and installation information, see Java SE Downloads.	
Default Database	Embedded H2 database (not recommended for production use)	Embedded PostgreSQL Database (not recommended for production use)
Supported Databases	MySQL 5.5, 5.6, 5.7 MariaDB 5.5	MySQL 5.5, 5.6, 5.7 MariaDB 5.5 PostgreSQL 8.1, 8.3, 8.4, 9.1, 9.2, 9.3, 9.4, 9.5

Note: In production environments, you should use an external MySQL or MariaDB database for Cloudera Director. For information on using an external MySQL database in place of the H2 embedded database, see Using MySQL for Cloudera Director Server on page 118. For information on using an external MariaDB database in place of the H2 embedded database, see Using MariaDB for Cloudera Director Server. By default, Cloudera Director stores its environment and cluster data in the embedded H2 database located at `/var/lib/cloudera-director-server/state.h2.db`. Back up this file to avoid losing the data. Cloudera strongly recommends using MySQL or MariaDB for production deployments of Cloudera Director, instead of H2. *Use of the H2 database in production environments can result in excessive space consumption for database files and slow database access. Unlike managed MySQL and MariaDB databases, H2 files are not backed up regularly, which puts your production deployment of Director at risk of data loss.*

Note: The versions of PostgreSQL listed above are supported with Cloudera Manager and CDH 5.11. Setting up PostgreSQL via Amazon RDS for Cloudera Manager and CDH is *not* supported. For a table of PostgreSQL versions supported with earlier versions of Cloudera Manager and CDH, see the PostgreSQL section of CDH and Cloudera Manager Supported Databases in the Cloudera Enterprise release notes. For information on setting up external database servers and on creating databases on *existing* database servers, see Using an External Database for Cloudera Manager and CDH on page 143.

Note: To run Kafka and Sentry on the same cluster, you must use Kafka 2.1 with Cloudera Manager and CDH 5.9 or 5.10.

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

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

	Cloudera Director	Cloudera Manager and CDH
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: • STANDARD_D12_v2 • STANDARD_D13_v2 • STANDARD_D14_v2 • STANDARD_D15_v2 • STANDARD_DS12_v2 • STANDARD_DS13_v2 • STANDARD_DS14_v2 • STANDARD_DS13 • STANDARD_DS14 • STANDARD_DS15_v2 • STANDARD_GS4 • STANDARD_GS5

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

If you are using Cloudera Director 2.5 to deploy Cloudera Manager and CDH, the latest released version of Cloudera Manager 5.12 and CDH 5.12 is installed by default. To use any other version of Cloudera Manager or CDH, 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 50.
- For Google Cloud Platform, see Deploying Cloudera Manager and CDH on Google Compute Engine on page 64.
- For Microsoft Azure, see Deploying Cloudera Manager and CDH on Microsoft Azure.

Networking and Security Requirements

Cloudera Director recommends 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	*security_group_id* See note paragraph below.

Note: In AWS, 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.

In a restricted network environment, you may want to enable minimal network traffic between instances and keep open ports to a minimum rather than enabling all network traffic between cluster instances. For information about minimal port requirements, see Ports Used by Cloudera Director.

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), Google Cloud Platform, and Microsoft Azure.

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. AWS only supports Spot instances on CentOS, not on RHEL.

For more information about using Spot instances with Cloudera Director, see Using Spot Instances.

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**. Make sure that **DNS Hostnames** is set to **Yes** in the **Edit DNS Hostnames** dialog.
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`	*security_group_id*
`SSH (22)`	`TCP (6)`	22	*<your ip address>*

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.

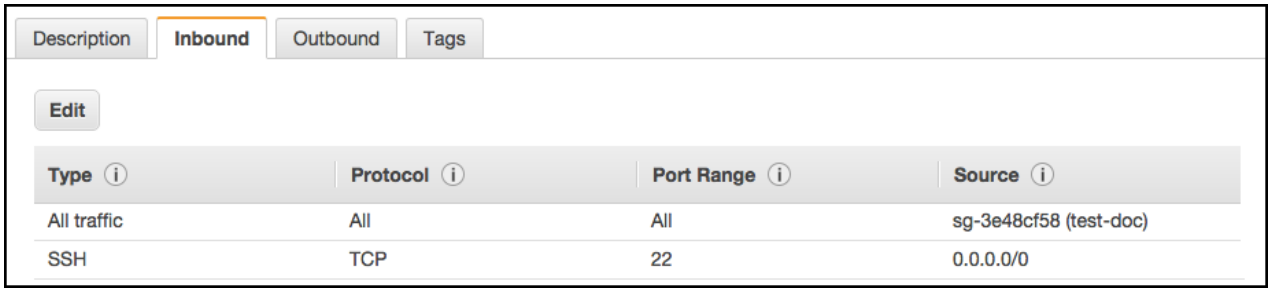
Creating a New Security Group

The simplest way to set up the required network connectivity for Cloudera Director is to create a security group for your VPC and allow traffic between members of this security group as described below. With this approach, you do not have to specify each part that is required by Cloudera Manager.

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.



For more information about security groups in AWS, see Security Groups for Your VPC. If your organization's network policies are more restrictive, and you need to specify each port required by Cloudera Manager, see Ports Used by Cloudera Manager and Cloudera Navigator in the Cloudera Manager documentation for details.

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. For Mac OS X, the key pair file is initially stored in the **Downloads** folder.
   - On Mac OS X, a secure location for storing the private key file is the hidden `~/.ssh` folder.
   - Enter the following command in a terminal window to move the key pair file from the **Downloads** folder to the SSH folder:

```
$ mv name_of_key_pair.pem ~/.ssh
```

You are now ready to launch an EC2 instance.

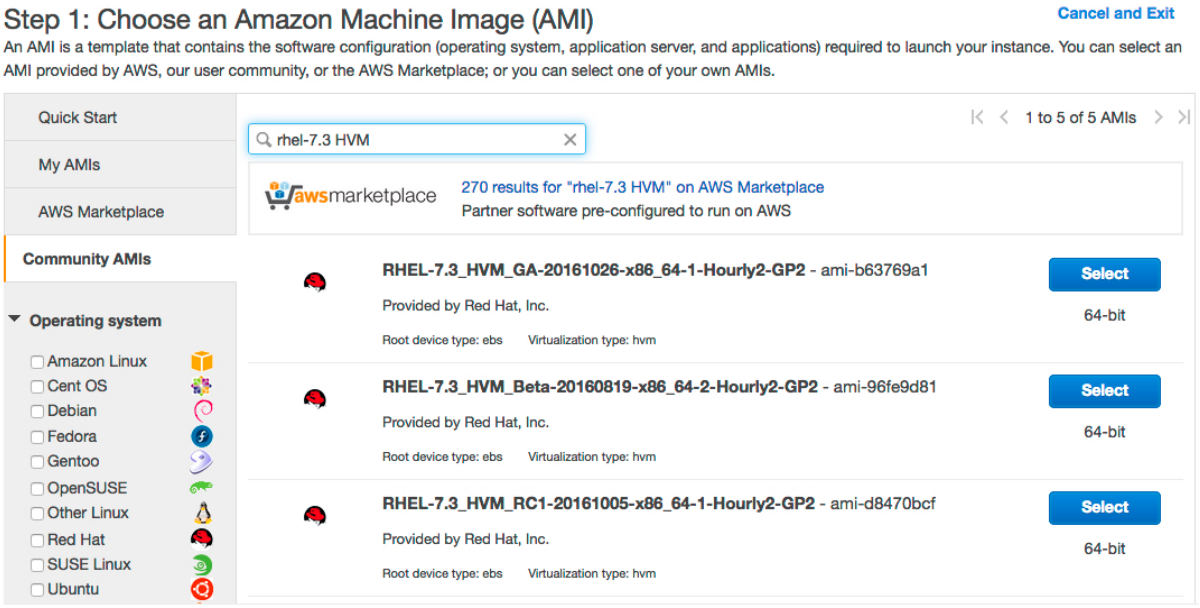
Launching an EC2 Instance for Cloudera Director

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.

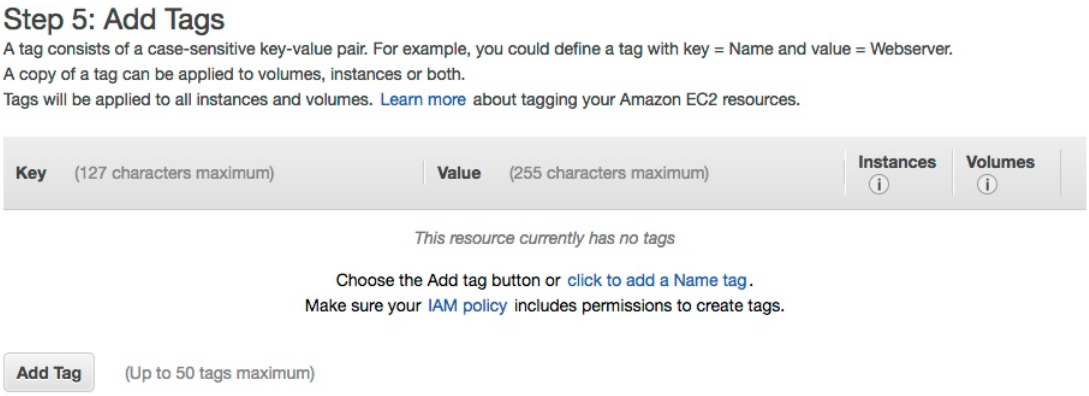
> **Note:** Alternatively, you can install Cloudera Director in a different region, on a different cloud provider, or a different network environment. For information on these more complex setups, see Running Cloudera Director and Cloudera Manager in Different Regions or Clouds on page 108.

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 37.
   a. Select **Community AMIs** in the left pane.
   b. In the search box, type the desired operating system. For example, if you type `rhel-7.3 HVM`, the search results show the versions of RHEL v7.3 that support HVM. Select the highest GA number to use the latest release of RHEL v7.3 supporting HVM.



   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: Add Tags**. On the Add Tags page, click the **Add Tag** button. For the **Name** key, enter a name for the instance in the **Value** field. Optionally, click **Add Tag** again to create an additional tags for the instance (up to a maximum of 50 tags).

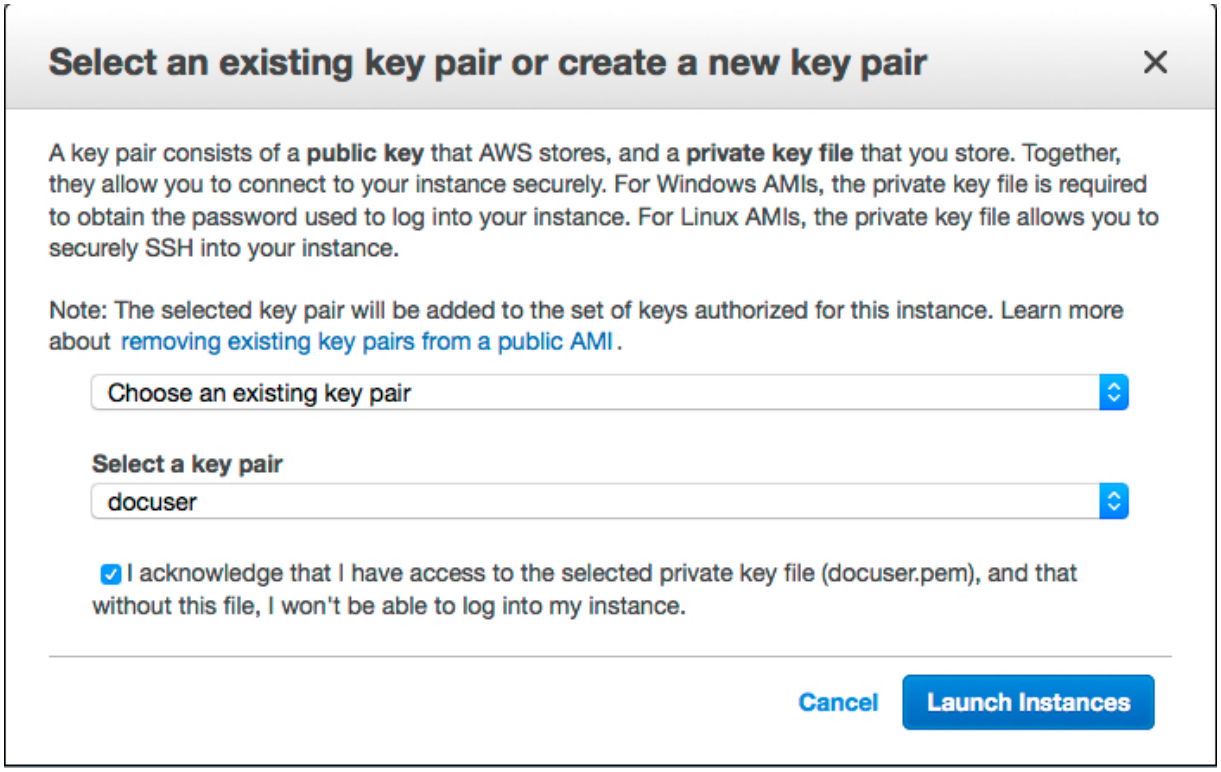


8. Click **Next: Configure Security Group**.
9. On the **Configure Security Group** page, create a new security group or add ports to an existing group. (If you have already created a security group with the required ports for Cloudera Director, as described on the previous page Setting up the AWS Environment, 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	*<your ip address>*
ALL Traffic	ALL	ALL	*security_group_id*

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 41.
    b. Click the check box to acknowledge that you have access to the private key.



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.

> **Note:** Cloudera strongly recommends using MySQL for production deployments of Cloudera Director, instead of H2. Use of the H2 database in production environments can result in excessive space consumption for database files and slow database access.

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
```

> **Note:** Depending on the operating system, different AMIs may require different SSH logins. To check the SSH login name for your EC2 instance, go to the **Instances** page in the AWS Console and click the instance name to display details for your instance and click the **Usage Instructions** tab. In this example, `centos` is used, not ec2-user:
>
>
>

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 on the Oracle web site. 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 include `wget`, 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.

> **Note:** Spark 2.2 requires JDK 8 and Python 2.7 or higher. For information on using Java 8 with Cloudera Director, see Deploying a Java 8 Cluster.

   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. 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
```

3. Add the signing key:

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

4. 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 JDK 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
```

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.

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`.
- 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, Cloudera recommends that you not configure security groups to allow internet access to your instances on the instances' public IP addresses. Instead, 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. The SOCKS proxy provides you with access to the Cloudera Director UI, Cloudera Manager UI, Hue UI, and any other cluster web UIs without exposing their 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 terminal process is later terminated.
- *your-key-file*.pem is the private key you used to create the EC2 instance where Cloudera Director is running.
- 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.)
- `ec2-user` is the AMI username for the EC2 instance where Cloudera Director is running. The AMI username can be found in the details for the instance displayed in the **AWS Management Console** on the **Instances** page under the **Usage Instructions** tab.
- *instance_running_director_server* is the private IP address of the EC2 instance running Cloudera Director server, if your networking configuration provides access to it, or its public IP address if not.
- & (optional) causes the SSH connection to run as an operating system background process, independent of the command shell. (Without the &, you 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).

Step 2: Configure Your Browser to Use the Proxy

On Google Chrome

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

- The SOCKS proxy port to use (must be the same value used in step 1)
- The profile to use (this example creates a new profile)

This creates a new profile and launches a new instance of Chrome that does not interfere with any currently running instance.

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 fully qualified domain name (FQDN). For example, when you connect to the Cloudera Director server, Cloudera Manager server, or Hue UI server, the browser actually connects to the proxy server, which performs the SSH tunneling.

Setting Up SwitchyOmega on the Google Chrome Browser

If you use Google Chrome, and especially if you use multiple proxies, the SwitchyOmega browser extension is a convenient tool to configure and manage all of your proxies in one place and switch 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:

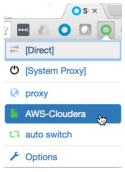
```
function regExpMatch(url, pattern) {
  try { return new RegExp(pattern).test(url); } catch(ex) { return false; }
}

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.



You are now ready to deploy Cloudera Manager and CDH.

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. For more information, see Using Custom Repositories with Cloudera Manager and CDH on page 107.

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 42. Include port 7189 in the address. For example:

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

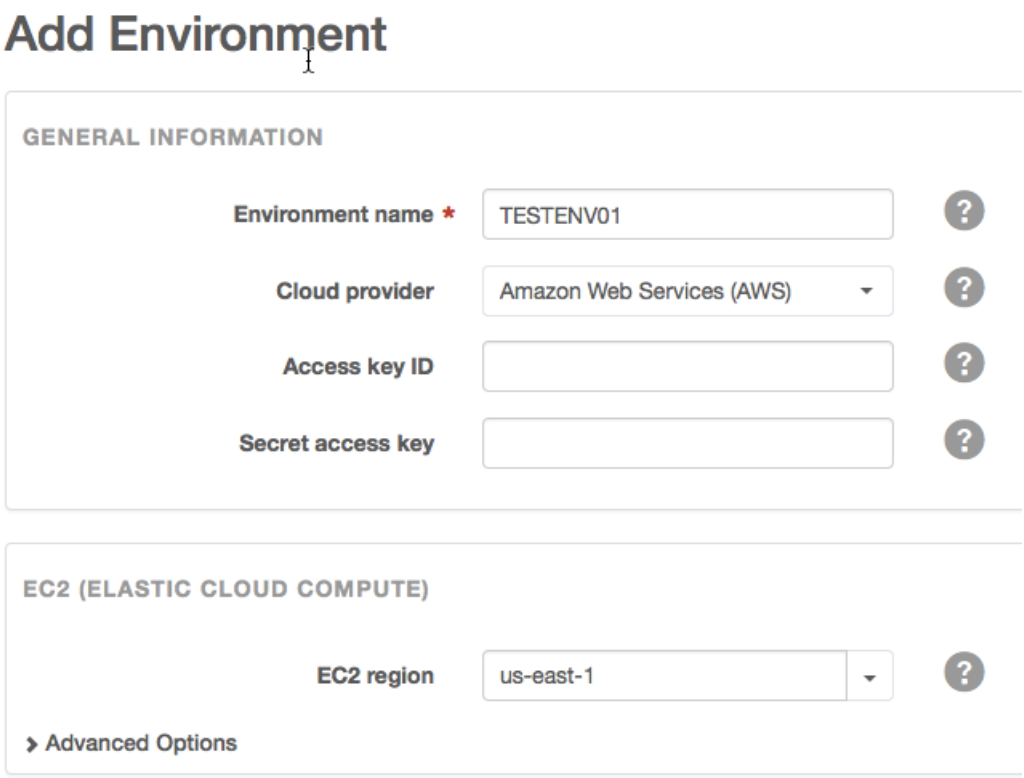
2. In the **Cloudera Director** login screen, enter `admin` in both the **Username** and the **Password** fields.
3. In the Cloudera Director **Welcome** screen, click **Let's get started**.

   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.

      > **Note:** Leave the **Access key ID** and **Secret access key** fields blank if you are using AWS Identity and Access Management (IAM) for authentication and authorization.

   d. In the **EC2 region** field, select the same region in which your Cloudera Director instance was created.

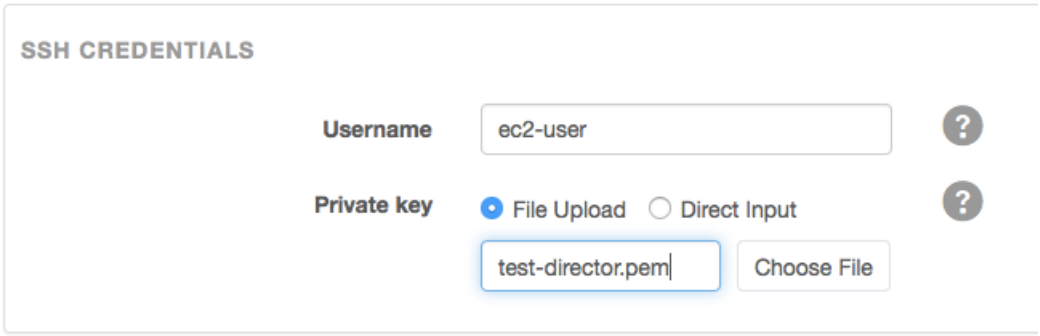


e. In the **SSH Credentials** section:

a. For a RHEL AMI, enter **ec2-user** in the **Username** field.

> **Note:** Depending on the operating system, different AMIs may require different SSH logins. To check the SSH login name for your EC2 instance, go to the **Instances** page in the AWS Console and click the instance name to display details for your instance and click the **Usage Instructions** tab. In this example, `centos` is used with a Centos AMI, not ec2-user:
>
>
>

b. Copy the SSH private key you created in Launching an EC2 Instance for Cloudera Director on page 42 in the **Private key** field.



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, 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:
   a. In the **Instance Template name** field, enter a name for the template.
   b. In the **Instance type** field, select **m4.large** or **m4.xlarge**.
   c. 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 42, or find another AMI with a supported operating system.

      > **Note:** To reduce cluster bootstrap times, you can preload the AMIs you use with Cloudera Manager and CDH. For more information, see Using Custom Repositories with Cloudera Manager and CDH on page 107.

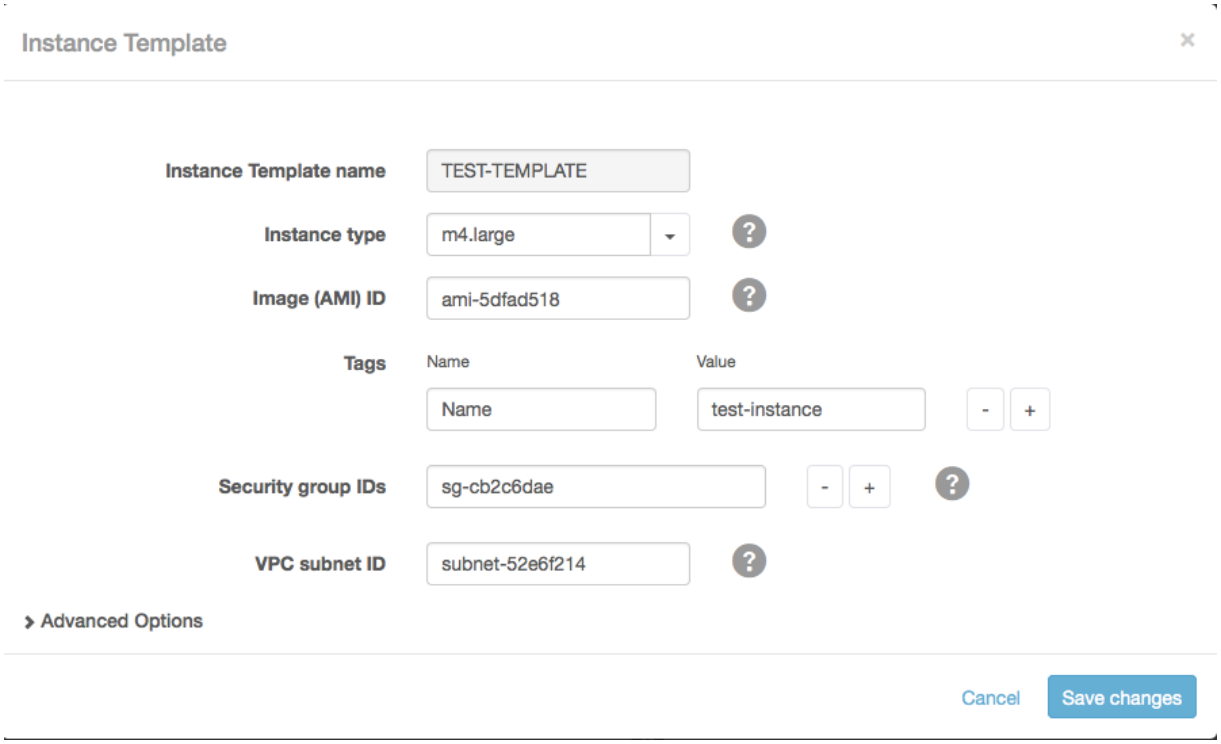
   d. In the **Tags** field, add one or more tags to associate with the instance.

      > **Note:** Cloudera Director uses the **Name** field on AWS. By default, Cloudera Director puts the following values in the **Name** field: Cloudera Director id, Cloudera Director template name. If you want to create a custom key value for this field in place of "Name," see Configuring Cloudera Director to Use Custom Tag Names on AWS.

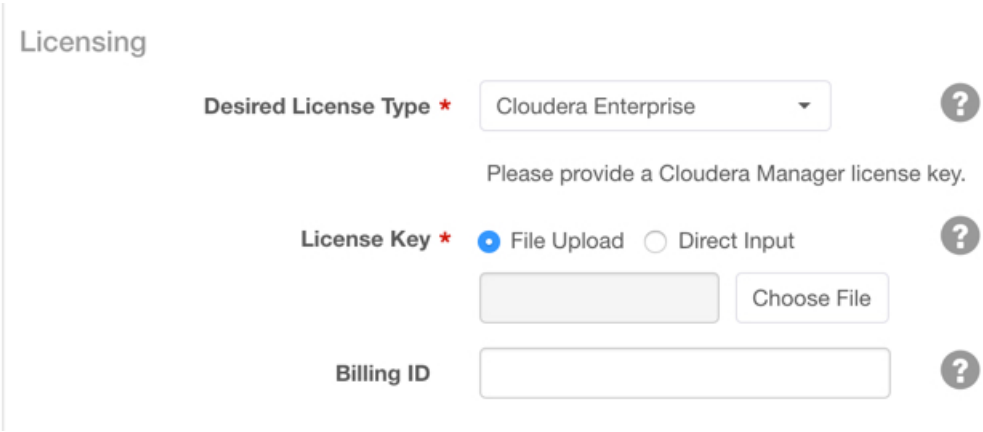
   e. In the **Security group IDs** field, enter the security group ID you set up in Creating a New Security Group on page 42.
   f. In the **VPC subnet ID** field, enter the ID of the VPC subnet that was created during VPC setup.
   g. Click **Advanced Options** if you want to specify additional values for optional features, such as EBS, IAM, Spot instances, and AWS user data.

      > **Note:** With Cloudera Director 2.4 and higher, you can enter opaque data or scripts in the **User data** field of the instance template, and this data will be passed to the instances as they are launched, using the EC2 user data mechanism. The data will be base64-encoded automatically. For more information, including examples, see Instance Metadata and User Data in the AWS documentation.

   h. Click **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.

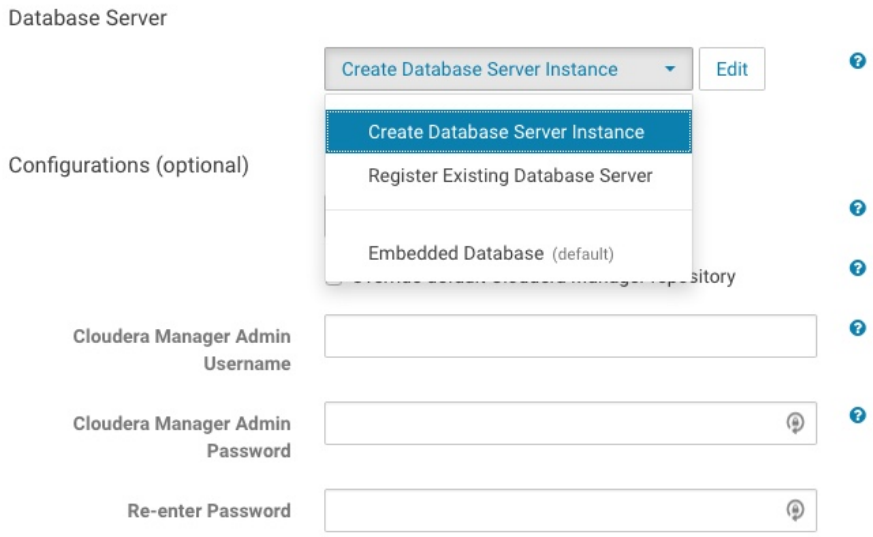


   To enable usage-based billing, you must have a Cloudera Enterprise license and a billing ID provided by Cloudera. Perform these steps in the **Add Cloudera Manager** screen:

   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 **Database Server** section, the default selection is **Embedded Database**. This installs an embedded PostgreSQL database for Cloudera Manager. The embedded PostgreSQL database should be used only when creating a demonstration or proof-of-concept deployment. It is *not recommended* for production. If desired, select **Create**

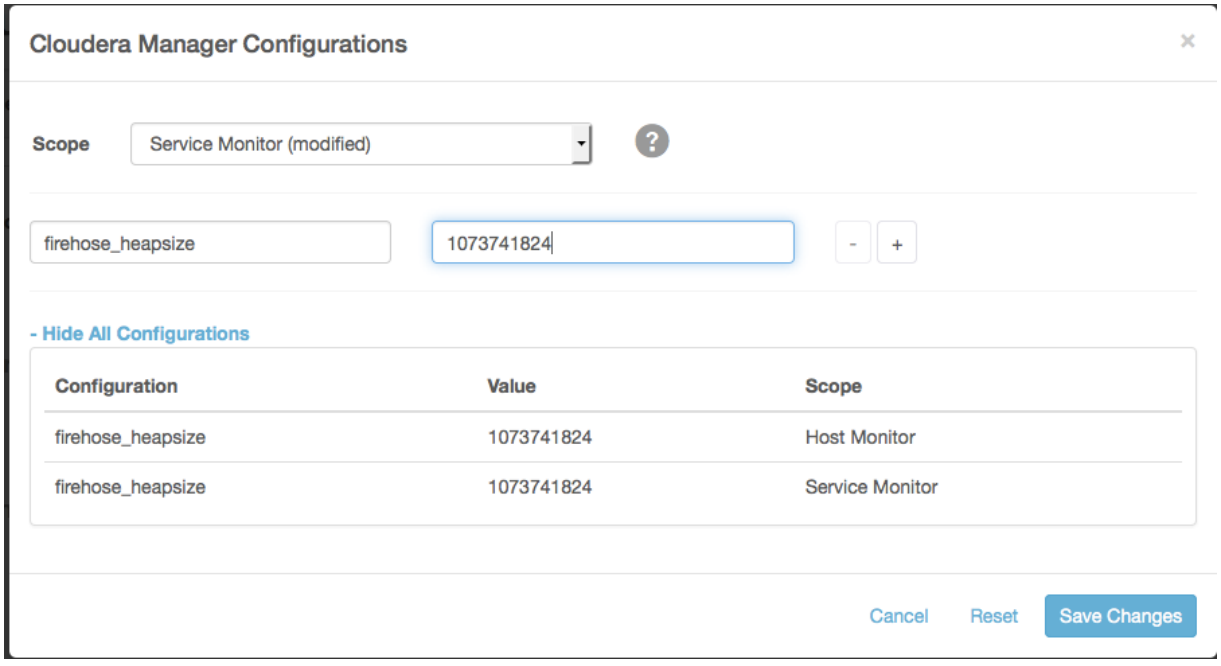
Database Server Instance or **Register Existing Database Server** from the dropdown list, instead of **Embedded Database**:



For information about using an external database for Cloudera Director server, see the following pages:

- Using MySQL for Cloudera Director Server on page 118
- Using MariaDB for Cloudera Director Server on page 131

10. In the **Add Cloudera Manager** screen, click **Cloudera Manager Configurations**.
11. 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**.



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

Cloudera Director version	Cloudera Manager version installed
Cloudera Director 2.0	Latest released version of Cloudera Manager 5.5
Cloudera Director 2.1	Latest released version of Cloudera Manager 5.7
Cloudera Director 2.2	Latest released version of Cloudera Manager 5.8
Cloudera Director 2.3	Latest released version of Cloudera Manager 5.10
Cloudera Director 2.4	Latest released version of Cloudera Manager 5.11
Cloudera Director 2.5	Latest released version of Cloudera Manager 5.12

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, the repository URL for Cloudera Manager 5.5.4 on any supported version of RHEL 7 is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.5.4/.

> **Note:** The Cloudera Manager minor version must 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 RHEL 7 is http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera.

13. In the **Add Cloudera Manager** screen, click **Continue**.
14. At the **Confirmation** prompt, click **OK** to begin adding a cluster.
15. 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:

Cloudera Director version	CDH version installed
Cloudera Director 2.0	Latest released version of CDH 5.5
Cloudera Director 2.1	Latest released version of CDH 5.7
Cloudera Director 2.2	Latest released version of CDH 5.9
Cloudera Director 2.3	Latest released version of CDH 5.10
Cloudera Director 2.4	Latest released version of CDH 5.11
Cloudera Director 2.5	Latest released version of CDH 5.12

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, choose an existing instance template or create a new one, either for the all instance groups in the cluster, or for each group. For each instance group, indicate the number of instances you want.

> **Note:** To reduce cluster bootstrap times, you can preload the AMIs you use for your cluster instance groups. For more information, see Using Custom Repositories with Cloudera Manager and CDH on page 107.

If you want to use Spot instances for your **workers** group:

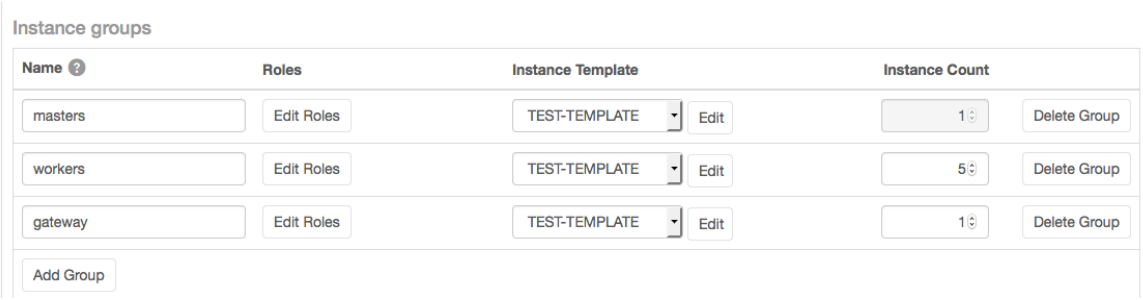
a. In the **Create New Instance Template** modal screen, click **Advanced Options**.

b. In the **Spot bid (USD/hr)** field, enter your Spot bid price.

c. Click the **Use Spot instances** checkbox.

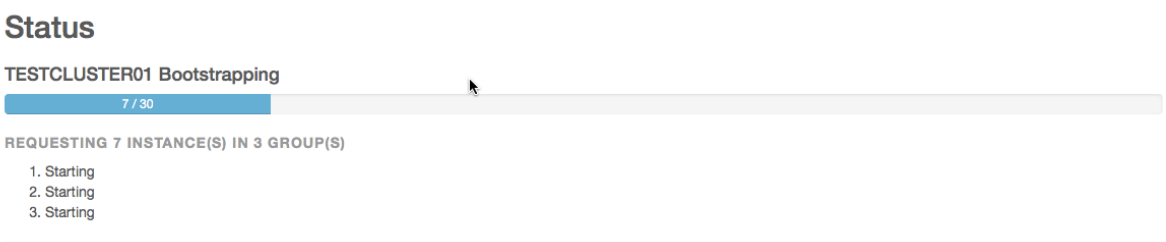
d. Click **Save Changes**.

For more information about using Spot instances with Cloudera Director, see Using Spot Instances on page 102.



16. Click **Continue**.

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



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

You are finished with the deployment tasks.

Pausing a Cluster in AWS

If all data for a cluster is stored on EBS volumes, you can pause the cluster and stop your AWS EC2 instances during periods when the cluster will not be used. The cluster will not be available while paused and can't be used to ingest or process data, but you won't be billed by Amazon for the stopped EC2 instances. Provisioned EBS storage volumes will continue to accrue charges.

> **Important:** Pausing a cluster requires using EBS volumes for all storage, both on management and worker nodes. Data stored on ephemeral disks will be lost after EC2 instances are stopped.

Shutting Down and Starting Up the Cluster

In the shutdown and startup procedures below, some steps are performed in the AWS console and some are performed in Cloudera Manager:

- For AWS actions, use one of the following interfaces:
  - AWS console
  - AWS CLI
  - AWS API
- For cluster actions, use one of the following interfaces:
  - The Cloudera Manager web UI
  - The Cloudera API **start** and **stop** commands

Shutdown procedure

To pause the cluster, take the following steps:

1. Stop the cluster (in Cloudera Manager).
2. Stop the Cloudera Management Service (in Cloudera Manager).
3. Stop all cluster EC2 instances, including the Cloudera Manager host (in AWS).

Startup procedure

To restart the cluster after a pause, the steps are reversed:

1. Start all cluster EC2 instances (in AWS).
2. Start the Cloudera Management Service (in Cloudera Manager).
3. Start the cluster (in Cloudera Manager).

More information

For more information about stopping the Cloudera Management Service, see Stopping the Cloudera Management Service in the Cloudera Enterprise documentation.

For more information about restarting the Cloudera Management Service, see Restarting the Cloudera Management Service in the Cloudera Enterprise documentation.

For more information about starting and stopping a cluster in Cloudera Manager, see Starting, Stopping, Refreshing, and Restarting a Cluster in the Cloudera Enterprise documentation.

For more information about stopping and starting EC2 instances, see Stop and Start Your Instance in the AWS documentation.

Considerations after Restart

Since the cluster was completely stopped before stopping the EC2 instances, the cluster should be healthy upon restart and ready for use. You should be aware of the following about the restarted cluster:

- After starting the EC2 instances, Cloudera Manager and its agents will be running but the cluster will be stopped. There will be gaps in Cloudera Manager's time-based metrics and charts.

- EC2 instances retain their internal IP address and hostname for their lifetime, so no reconfiguration of CDH is required after restart. The public IP and DNS hostnames, however, will be different. Elastic IPs can be configured to remain associated with a stopped instance at additional cost, but it isn't necessary to maintain proper cluster operation.

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.

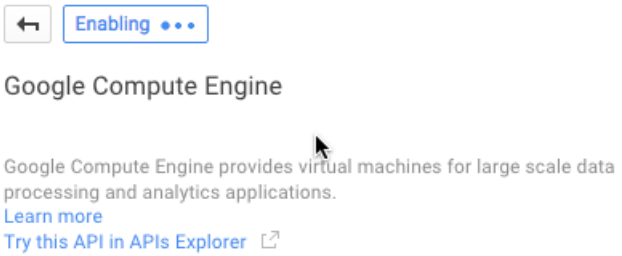
2. Create an SSH key.
3. Set up gcloud compute.

Creating a Service Account for Cloudera Director

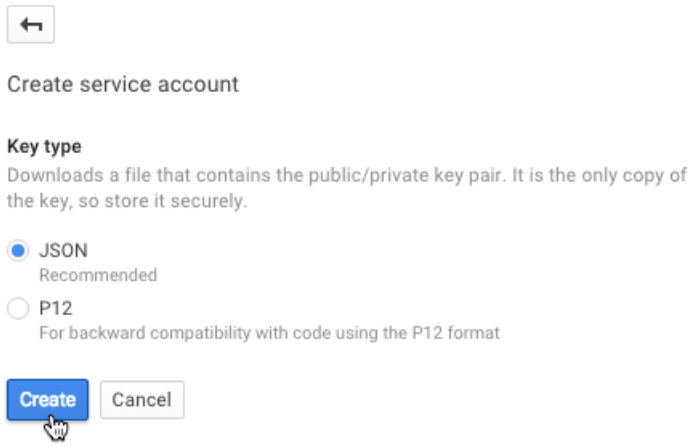
A service account enables Cloudera Director to authenticate to various Google Cloud Platform services, such as Google Cloud Storage. To create a service account, perform the following steps:

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.



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**.



   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

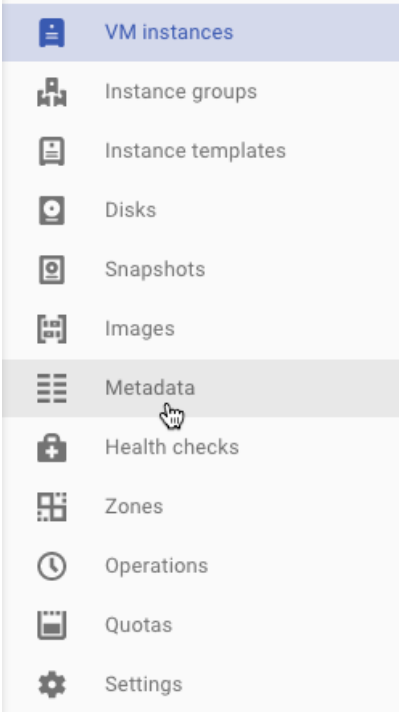
To SSH into an instance using your own terminal (as opposed to the Google Cloud Platform console), you must generate and upload an SSH key.

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.

Name	Description	Details/Restrictions
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 Cloud Platform 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 37.
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.

Note: Spark 2.2 requires JDK 8 and Python 2.7 or higher. For information on using Java 8 with Cloudera Director, see Deploying a Java 8 Cluster.

   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 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
```

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:

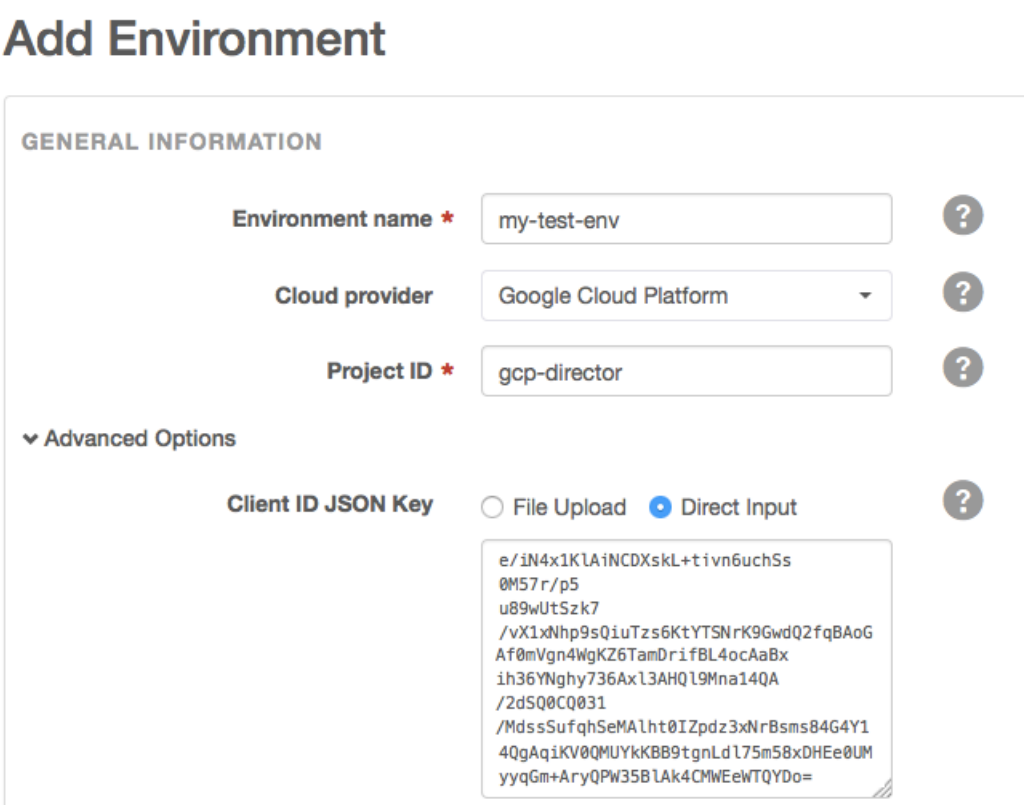
1. Open a web browser and go to the private IP address of the instance you created in Creating a Google Compute Engine VM Instance on page 60. Include port 7189 in the address. For example:

```
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.

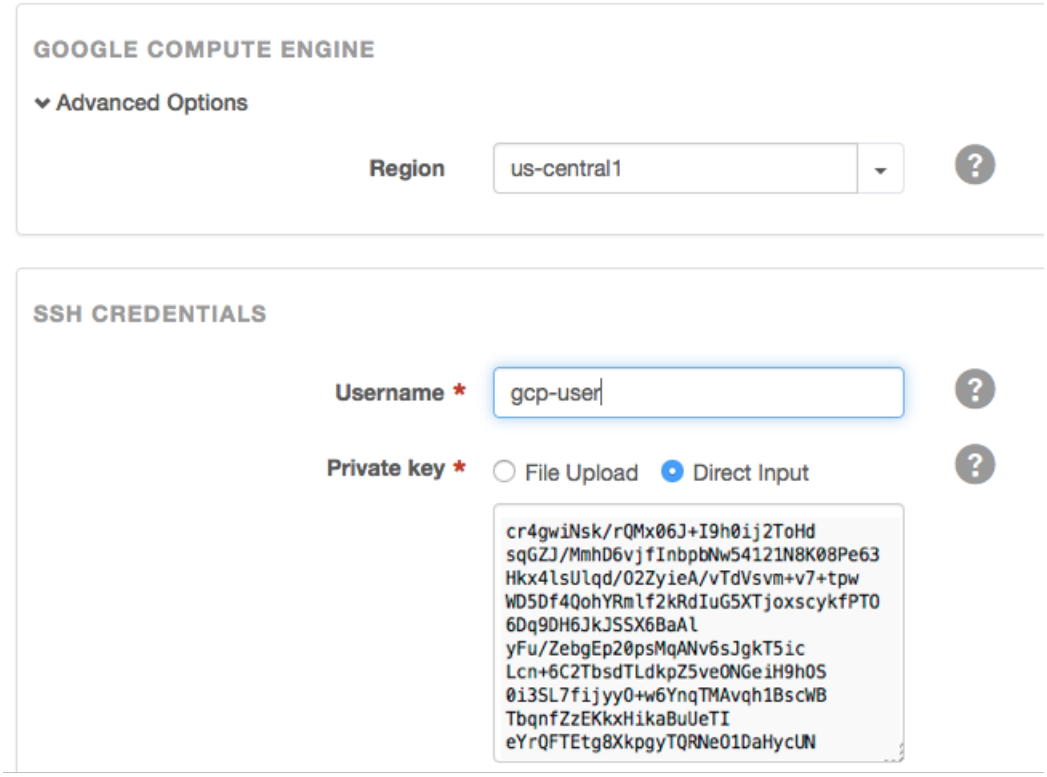
4. In the **Add Environment** screen:
   a. Enter a name in the **Environment Name** field.
   b. In the **Cloud provider** field, select **Google Cloud Provider**.
   c. In the **Project ID** field, enter the ID for the project you created in Creating a Google Cloud Platform Project on page 58.
   d. 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 58.



e. In the **Advanced Options** section, enter the same **region** that your Cloudera Director instance was created in.

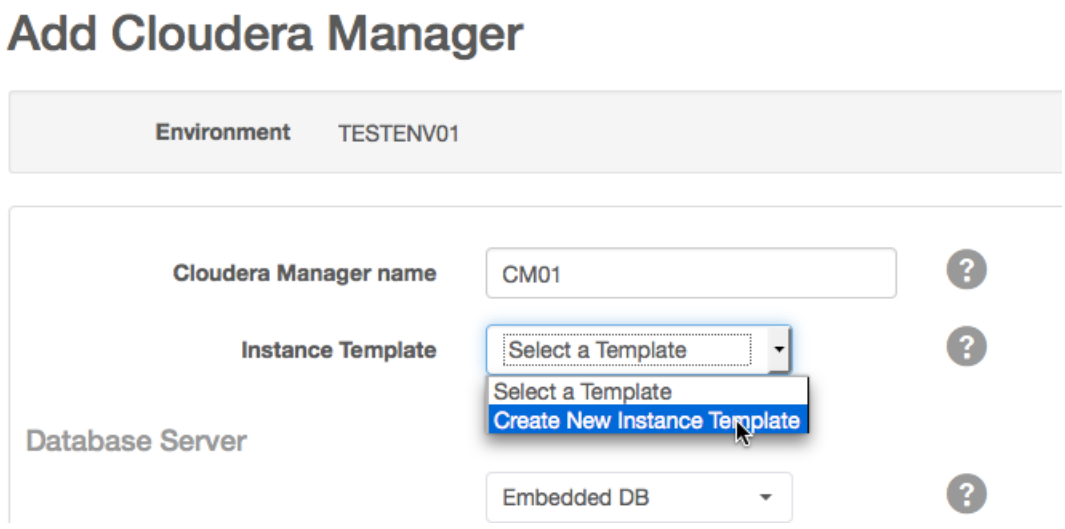
f. In the **SSH Credentials** section:

- 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 59 in the **Private key** field.



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.



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 60.

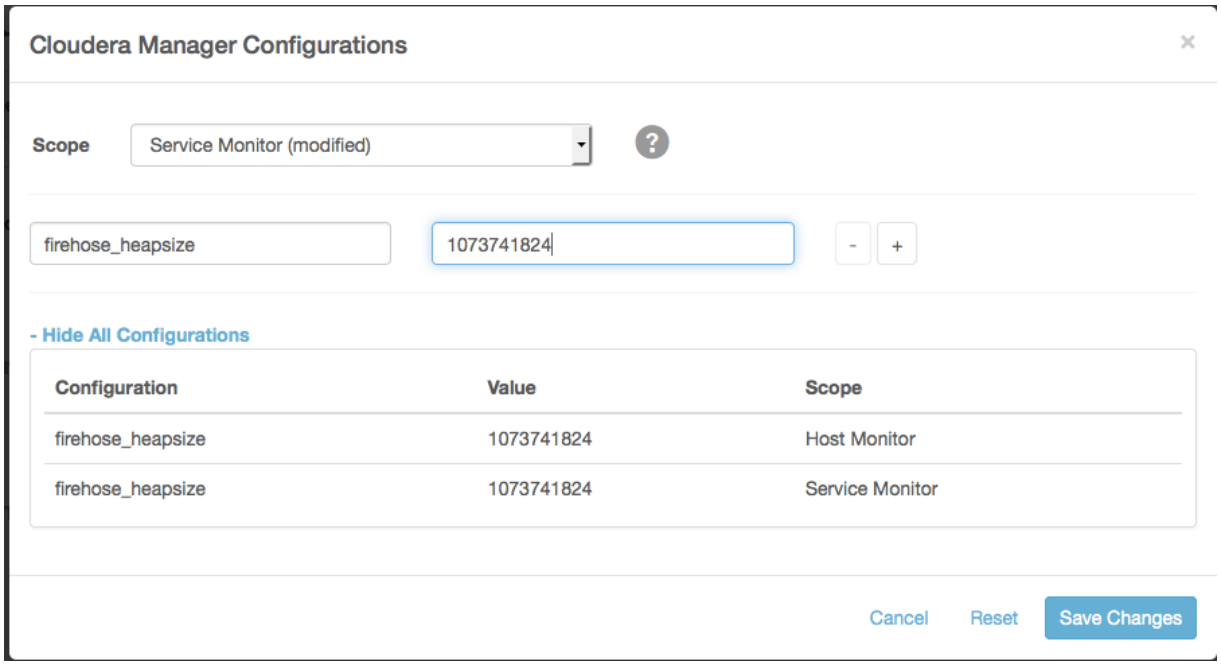
- In the **Tags** field, add one or more tags to associate with the instance.
- Click **Save changes**.

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**.



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

Cloudera Director version	Cloudera Manager version installed
Cloudera Director 2.0	Latest released version of Cloudera Manager 5.5
Cloudera Director 2.1	Latest released version of Cloudera Manager 5.7
Cloudera Director 2.2	Latest released version of Cloudera Manager 5.8
Cloudera Director 2.3	Latest released version of Cloudera Manager 5.10
Cloudera Director 2.4	Latest released version of Cloudera Manager 5.11
Cloudera Director 2.5	Latest released version of Cloudera Manager 5.12

   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 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.

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:

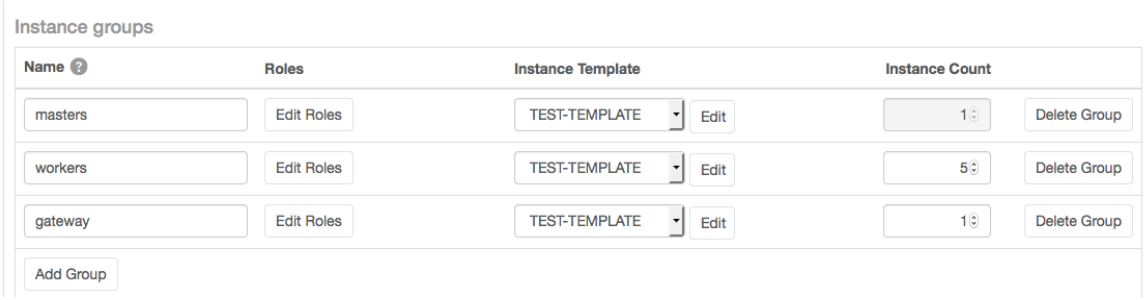
Cloudera Director version	CDH version installed
Cloudera Director 2.0	Latest released version of CDH 5.5
Cloudera Director 2.1	Latest released version of CDH 5.7
Cloudera Director 2.2	Latest released version of CDH 5.9
Cloudera Director 2.3	Latest released version of CDH 5.10
Cloudera Director 2.4	Latest released version of CDH 5.11
Cloudera Director 2.5	Latest released version of CDH 5.12

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.



14. Click **Continue**.
15. At the **Confirmation** prompt, click **OK** to deploy the cluster. Cloudera Director displays a status screen.

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 for creating them.

For best practices when creating a cluster on Microsoft Azure, see Cloudera Enterprise Reference Architecture for Azure Deployments.

Obtaining Credentials for Cloudera Director

To get started with Cloudera Director and Microsof t Azure, you create an Active Directory (AD) application and service principal and obtain the required Azure credentials for Director. 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.

Follow these general steps to obtain the required credentials:

1. Create the AD application and make sure that it has the **contributor** role in your Azure subscription, which allows you to create and delete resources. If you are not sure about these settings, contact your Active Directory administrator or Microsoft Azure Support.
2. Create the service principal. This is typically created by a system administrator or security administrator in your organization. This person must have administrator privileges for your Microsoft Azure subscription.
3. Obtain the following Azure credentials for Cloudera Director:
   - Subscription ID - You can get the subscription ID in the Azure Portal (either the new or old portal); see the Azure subscriptions blade.
   - Tenant ID
   - Client ID
   - Client Secret

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 (easier to follow and recommended)
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 topic describes how to set up various resources required by Microsoft Azure:

Setting Up Resource Groups

Cloudera Director requires you to set up resource groups that house the following cluster resources::

- Azure virtual machines (VMs)
- Azure virtual network (VNet)
- Azure network security group (NSG)

These resources typically have different lifecycles, so you may want to locate each in a separate resource group for convenience. For simplicity, you can locate them in 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.

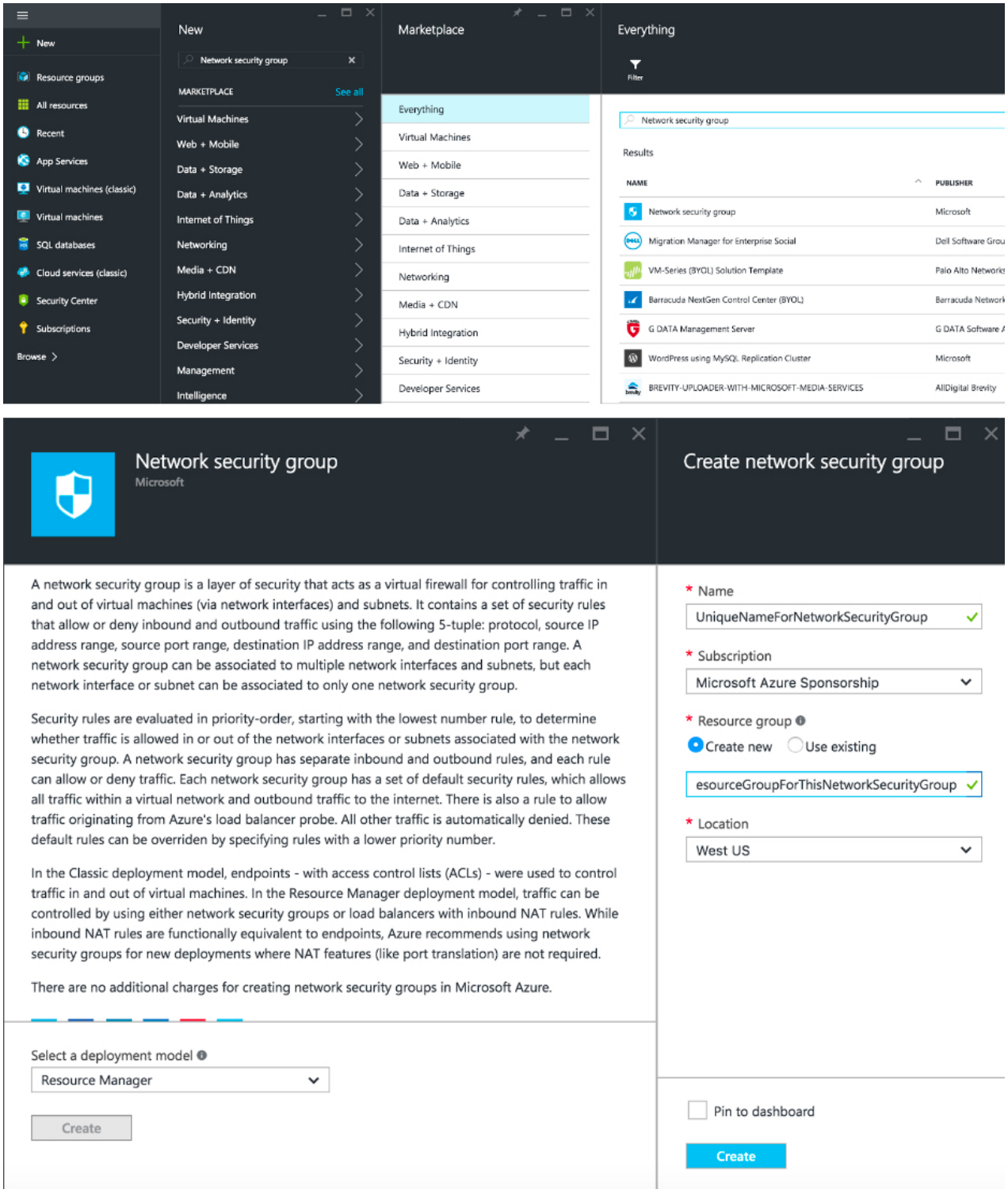
Note: You must open the following ports if you are allowing access to the web UI from public IP addresses:

- Cloudera Director - 7189
- Cloudera Manager - 7180

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.



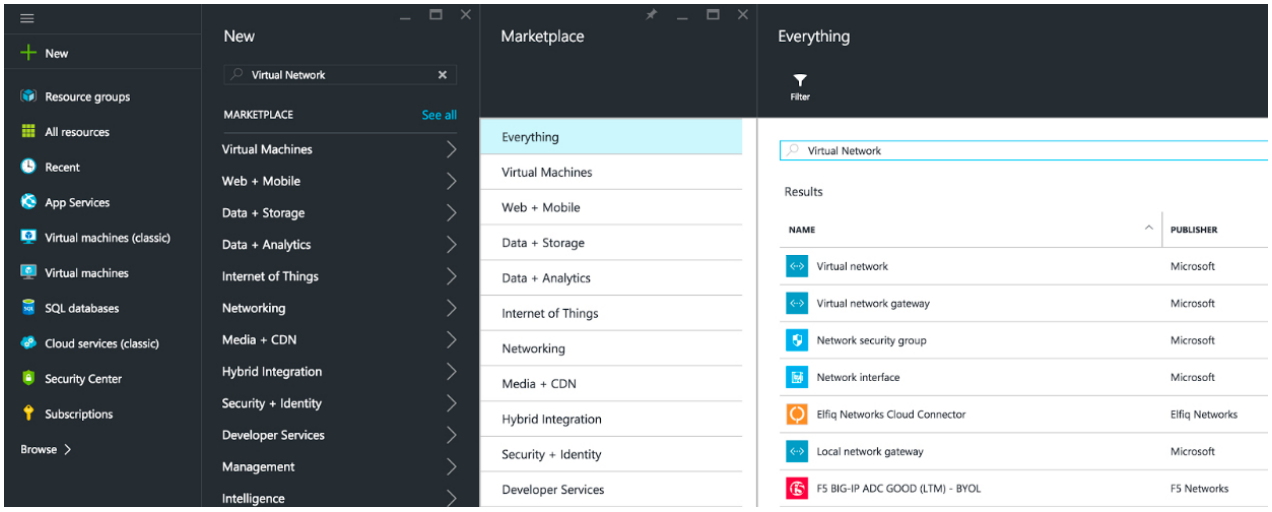
Setting Up a Virtual Network (VNet) and Subnet

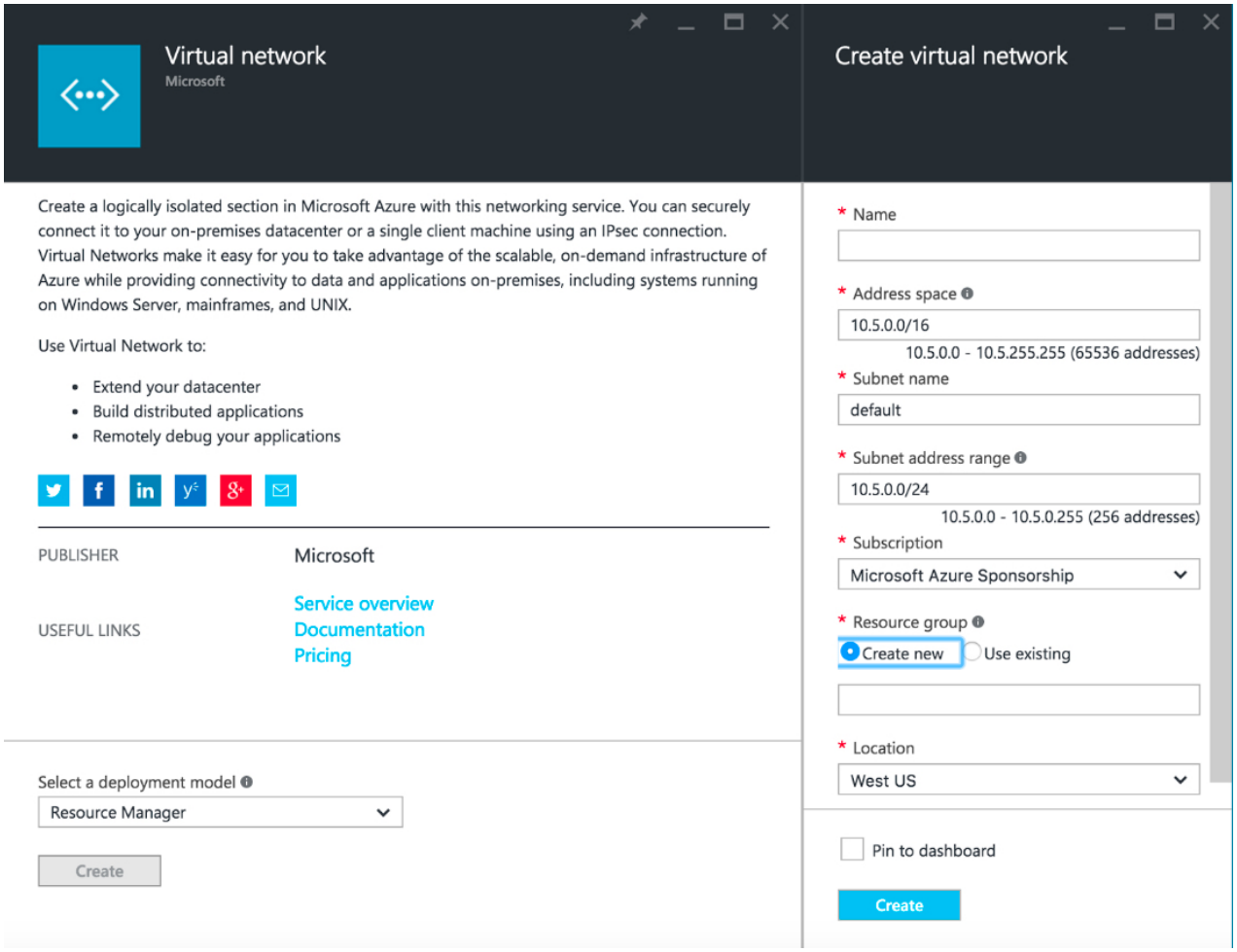
Cloudera Director requires a virtual network and subnet to implement its networking environment. The networking environment must be set up for forward and reverse hostname resolution. For a basic example for setting up forward and reverse hostname resolution, see Setting Up Dynamic DNS on Azure on page 75.

For an overview of virtual networks on Azure, see Virtual networks.

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

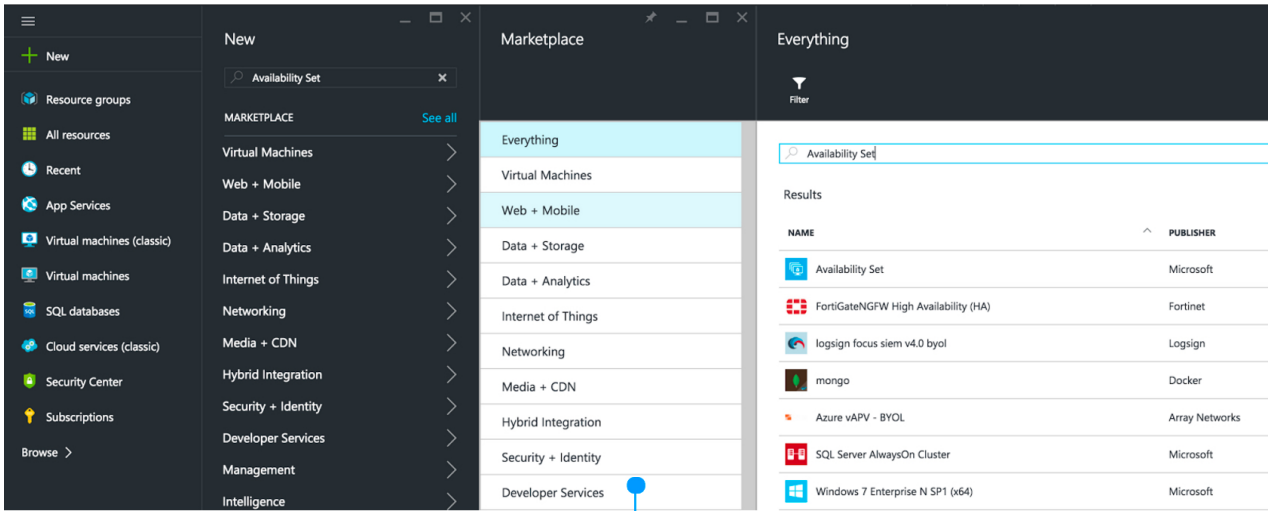
Azure uses availability sets to manage the availability of virtual machines. For best practices, in a CDH cluster, Cloudera recommends using one availability set for the master nodes and one availability set for the worker nodes. An availability set should not be shared by more than one CDH cluster.

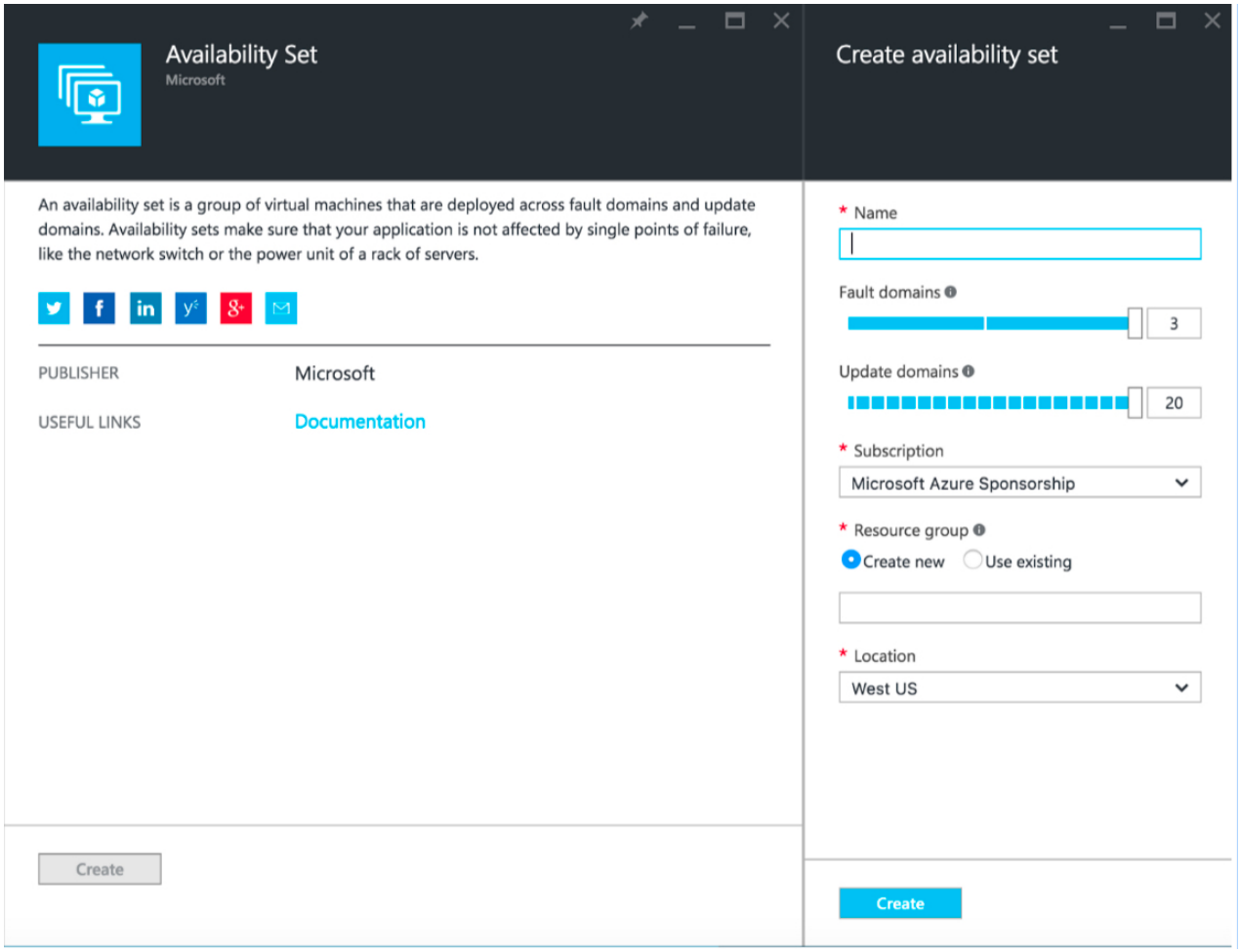
Read this Azure document for an overview of availability sets on Azure.

To create an availability set:

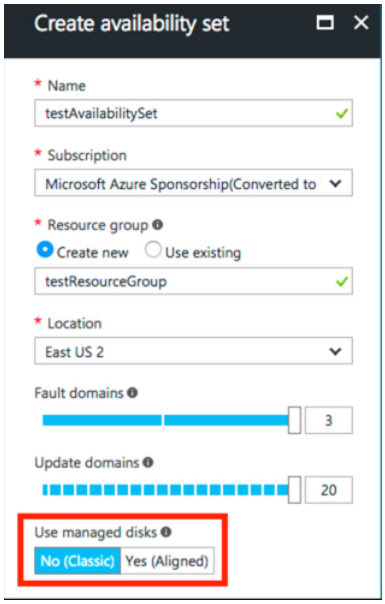
1. In the left pane, click **New**.
2. Type `Availability Set` in the search box.
3. Click **Availability Set** in the search result.
4. Click **Create**.
5. Type in a name for the availability set.
6. Type in a name for new resource group or select an existing resource group.
7. Increase the fault domain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

After creating the availability set for master nodes, repeat the steps to create an availability set for worker nodes.





When creating the Availability Set, make sure to select **No (Classic)** for the **Use managed disks** option. **No** is the default option.



Setting Up Dynamic DNS on Azure

This topic describes how to set up Dynamic DNS (DDNS) on Microsoft Azure.

Overview

Running Hadoop—specifically CDH, in this case—requires forward and reverse DNS for internal IP addresses, which is not currently supported in Microsoft Azure. 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. Following is basic example for setting up a 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.
- Creating required configuration and zone files.
- Creating update scripts that automatically update BIND when IP addresses are assigned or changed (for example, when stopping and starting hosts).

> **Note:** This document assumes certain configurations and architecture in some cases; those assumptions are noted.

The DNS Server and the Cloudera Director Host

Creating a DNS Server and Cloudera Director Host

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

Creating a Virtual Machine for the DNS Server

1. In Azure, select or create the resource group you will use for your cluster.
2. Select the **+** button to add a resource within that resource group.
3. 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 used for the DNS server.

Selecting DNS Defaults

Select an internal host fully qualified domain name (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:

The FQDN suffix you specify depends on your environment. Examples include `cdh-cluster.internal`, `cluster.company-name.local`, and `internal.company-name.com`.

> **Note:** Cloudera provides a set of scripts on the Cloudera GitHub site to automate the BIND install and setup process. You can use the scripts with CentOS 6.7 and 7.2 and RHEL 6.7 and 7.2. These 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. 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 required are included below. See the comments inline for changes you need to make. You must perform the following changes as root. 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 want to use for your cluster network. (This option is 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 exist.)

Starting BIND

1. chown `/etc/named*` to `named:named` (named requires read/write privileges):

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

2. Start BIND:

```
# service named start
```

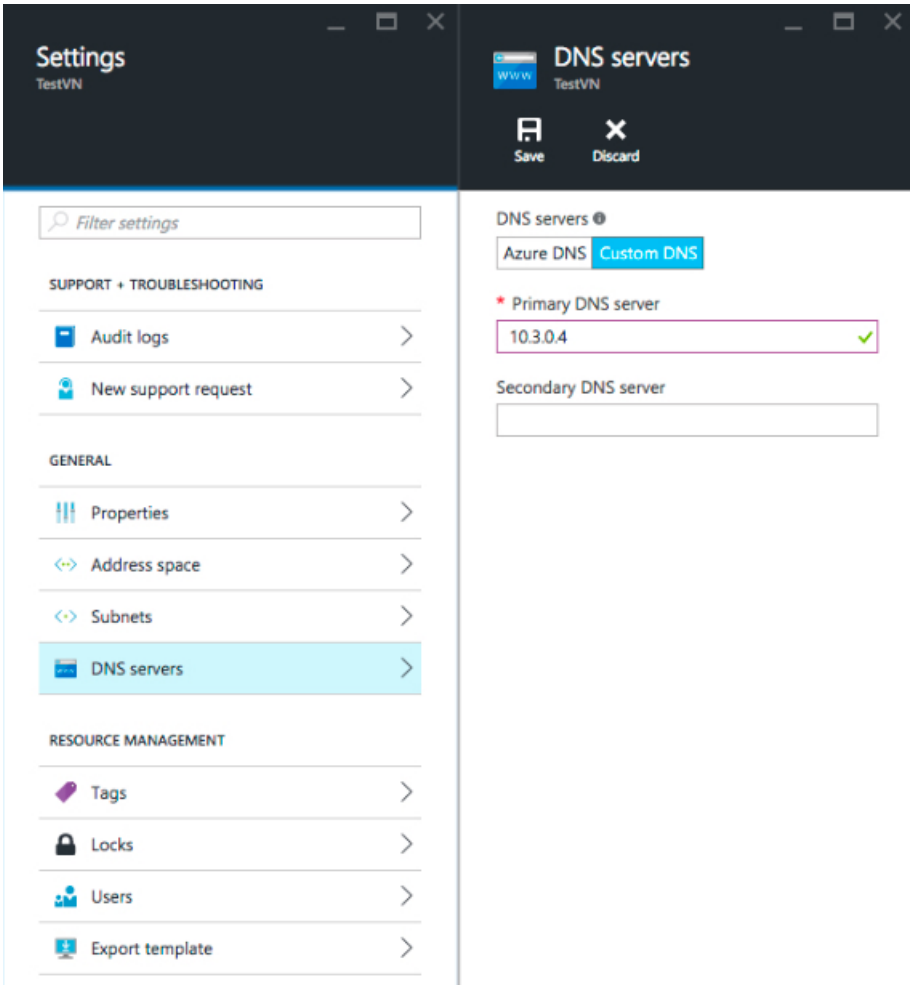
3. Set BIND to start on startup:

```
# chkconfig named on
```

Swapping DNS from Azure to BIND

To change the DNS settings on Azure:

1. In the left pane, click **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).



8. 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 automatically pick up the new DNS server address.

9. 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` reflects 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 creates a new `dhclient-exit-hooks` file in `/etc/dhcp/` and sets the file to be executable. Run the script as root:

```
#!/bin/sh
# cat a here-doc represenation 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 creates an `/etc/NetworkManager/dispatcher.d/12-register-dns` file and sets 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=$(mktemp -t nsupdate.XXXXXXXXX)
resolvconfupdate=$(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 the Azure hooks, and so can overwrite `/etc/resolv.conf` with custom values.

If you concatenate `/etc/resolv.conf`, it appears as follows:

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

You can now resolve internal FQDNs and perform 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 your implementation.

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. MySQL and PostgreSQL are supported.

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

Database Server Requirements

You can install a database server on the same host as Cloudera Director and DNS, or you can add one to a different host in the same virtual network as Cloudera Director and the cluster. Consider the following when installing a database server:

- It 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 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

For MySQL , follow the instructions in MySQL Database. Use the instructions for your specific version of MySQL, and keep in mind these addtional 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 (FQDN) that resolves to a private IP address.
- To reference the MySQL server by internal FQDN, make sure the MySQL server internal FQDN is registered with the DNS server you configured in Setting Up Dynamic DNS on Azure.

Setting Up a Virtual Machine for Cloudera Director Server

Cloudera Director server is used to provision CDH clusters. See Create a Linux VM on Azure using the Portal in the Azure documentation for an overview of creating a Linux VM on Azure. We recommend using the CentOS image published by Cloudera on Microsoft Azure Marketplace.

Consider the following when creating a Linux VM:

- 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 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, follow the procedure for the OS you use. You must be 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. Cloudera Director supports JDK versions 7 and 8.

   Note: Spark 2.2 requires JDK 8 and Python 2.7 or higher. For information on using Java 8 with Cloudera Director, see Deploying a Java 8 Cluster.

   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. 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

Sample configuration files are available on the Cloudera GitHub site. You can modify these configuration files to create clusters using the Cloudera Director CLI.

- azure.simple.conf: A simple Cloudera Director configuration that creates a Cloudera Manager node and a four-node cluster (one master and three workers).
- azure.reference.conf: A reference Cloudera Director configuration that creates an eight-node cluster (three masters and five workers) with high availability (HA) enabled.
- azure.kerberos.conf: The same Cloudera Director configuration as azure.reference.conf, 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 perform lookups directly from your Microsoft Azure network and allows you to connect to services using private IP addresses and internal fully qualified domain names (FQDNs).

This approach does the following:

- Sets 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.
- Changes the browser configuration to do all lookups through 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 are connecting from.
- The host that you proxy to must be able to reach the Cloudera Director server using a private IP. You can also proxy directly to the Cloudera Director server.

Start the SOCKS Proxy

To start a SOCKS 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

Google Chrome

By default, Google Chrome uses system-wide proxy settings on a per-profile basis. To start Chrome without these settings, use the command line and specify the following:

- The SOCKS proxy port ; this must be the same port you used when starting the proxy.
- The profile ; this example creates a new profile.

This create a new profile and launches a new instance of Chrome that does not conflict with any currently running Chrome instance.

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

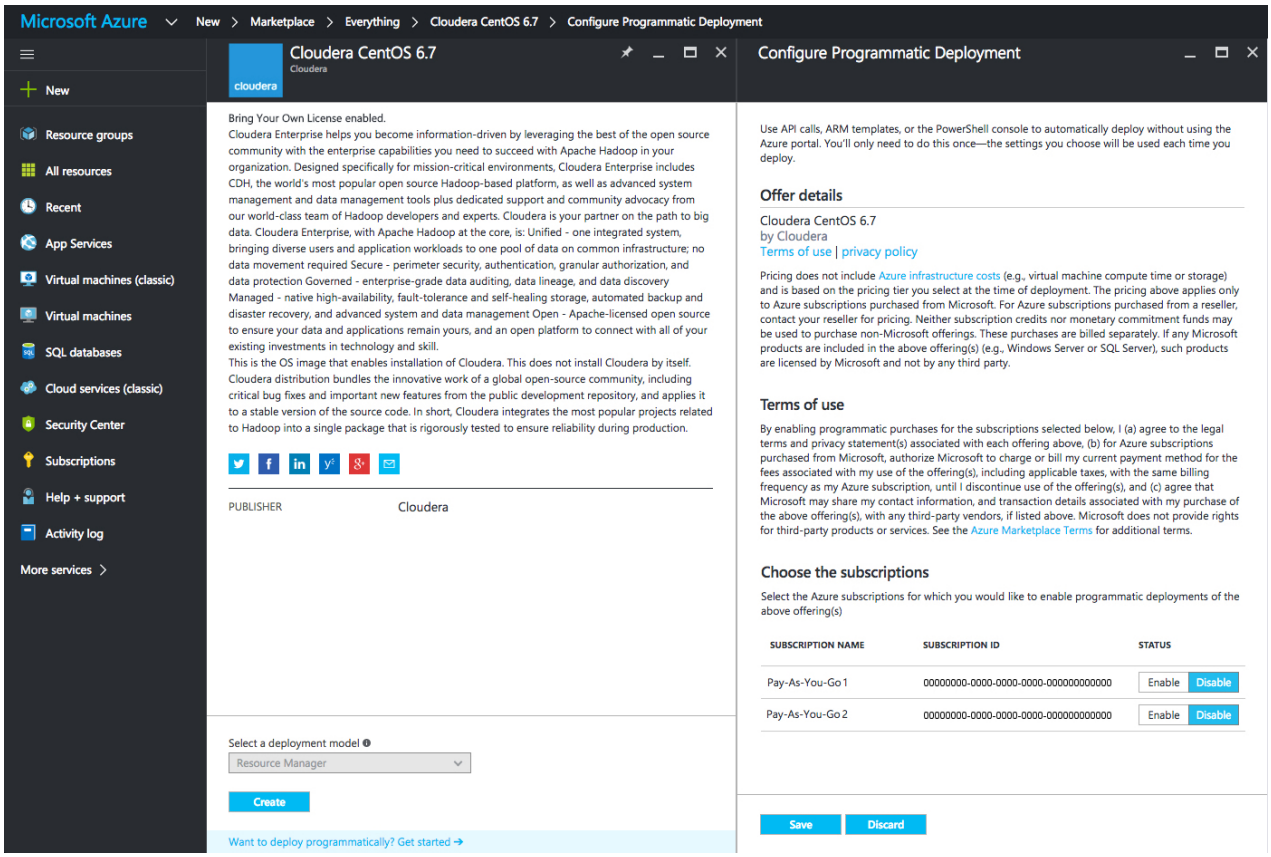
```
"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" ^
--user-data-dir="%USERPROFILE%\chrome-with-proxy" ^
--proxy-server="socks5://localhost:1080"
```

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 entering `localhost:7189` in the Chrome URL bar.

Allowing Access to VM Images

The Cloudera Director Azure plug-in deploys Azure VM images programmatically. To allow programmatic deployment of VM images on Azure, you must accept a term of usage and grant your Azure subscription permission to deploy the VM images.

By default, the Cloudera Director Azure plugin uses the Cloudera-certified CentOS 6 image. For detailed steps allowing programmatic deployment of Azure VM images, see aWorking with Marketplace Images on Azure Resource Manager.



Creating a Cluster

Before You Deploy Cloudera Manager and CDH

Important: Before using Cloudera Director to deploy clusters, 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.

This topic describes how to set up Cloudera Manager and a CDH cluster in Microsoft Azure using the Cloudera Director web UI. The following resources must be created and prerequisites must be met before beginning the deployment:

- An AD application and a service principal for the AD application. The AD application must have the **contributor** or similar role so that it has permission to create and delete resources in the subscription.

- A virtual network and network security group that is readily available for the cluster to use.
- The virtual network 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 to house cluster VMs.
- An availability set created in corresponding resource groups to house cluster VMs.
- Cloudera Director server VM.
- Cloudera Director server installed and running.
- Cloudera Director server access to the Azure virtual network (VNet).
- Database server that is readily available and 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

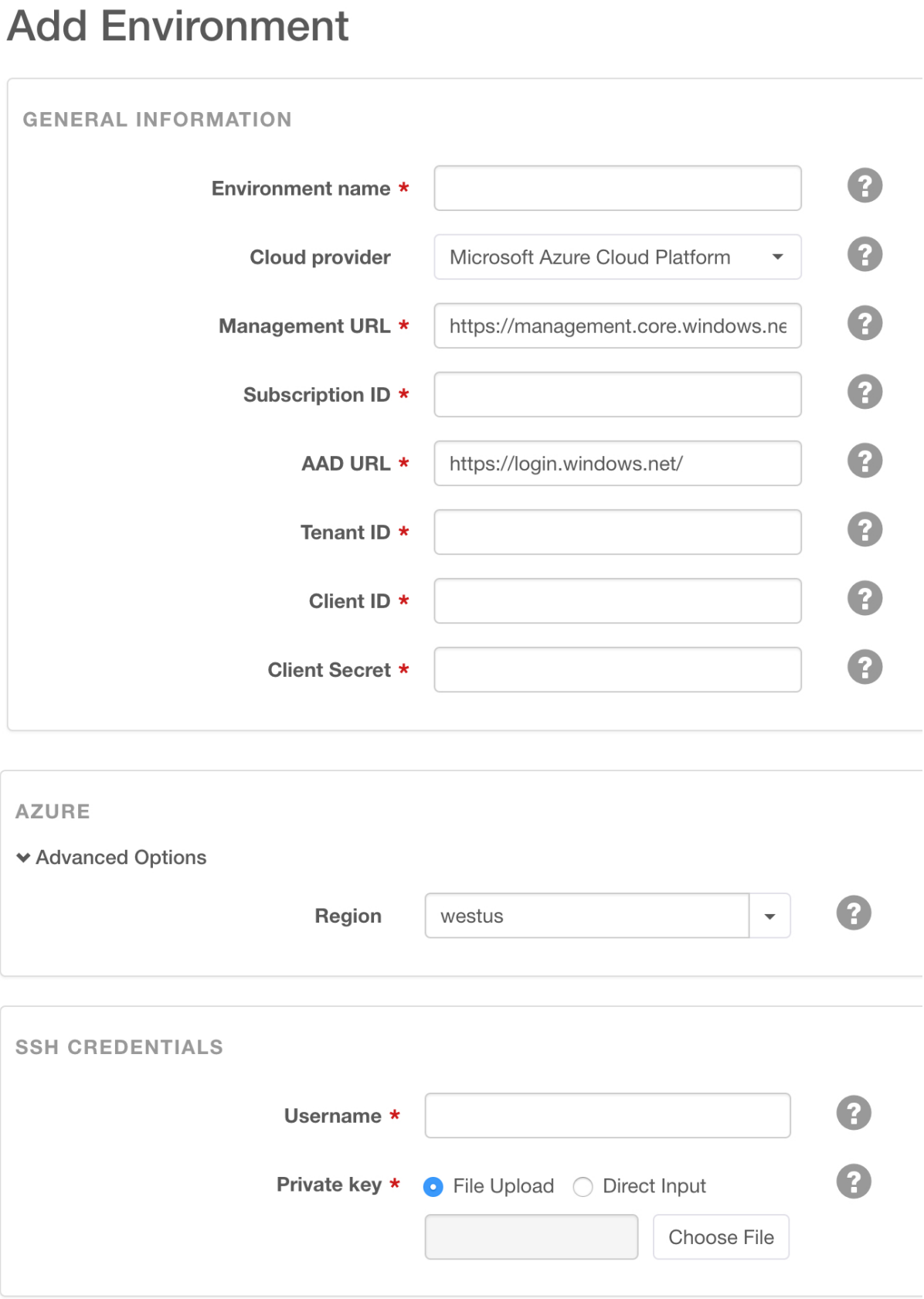
To deploy Cloudera Manager and CDH on an Azure VM instance, begin by creating an environment. The environment defines common settings, like region and key pair, that Cloudera Director uses with Azure. While creating an environment, you are also prompted to deploy its first cluster.

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, Cloudera Manager, and 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 do not 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 do not 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 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 creates 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.

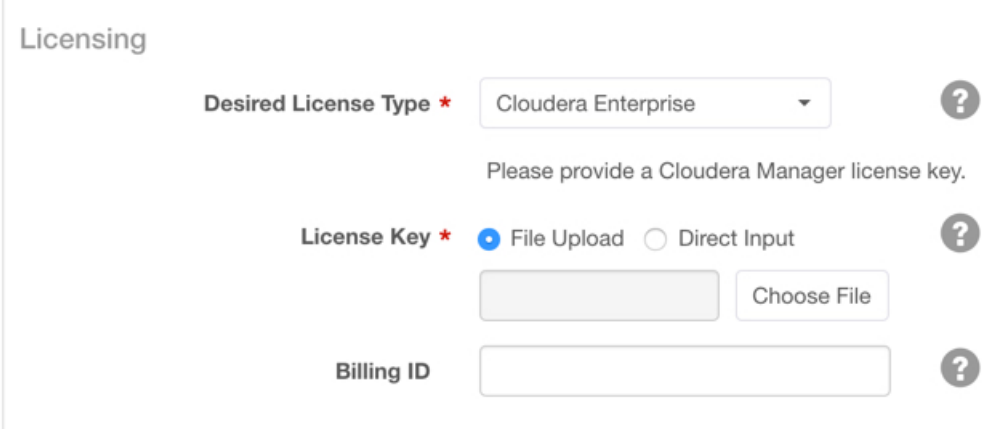


5. Click **Continue** to add Cloudera Manager.

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:
   a. In the **Instance Template** name field, enter a name for the template.
   b. In the **VirtualMachine Size** field, select one of the available sizes.
   c. In the **Image Alias** field, select one of the available images.
   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 want to use.
   i. In the **Host FQDN suffix** field, enter the name of the host FQDN suffix you want 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 **Storage Account Type** field, select **PremiumLRS**. For instance templates intended for worker nodes, you can select **StandardLRS**. See the Cloudera Reference Architecture for Microsoft Azure Deployments for details on supported storage account types and configurations.
   p. In the **Data Disk Count** field in **Advanced Options**, enter the number of data disks to attach for the VM.
   q. In the **Data Disk Size in GiB** field, select **1023**.
   r. Leave the **SSH username** field blank to use the username you set at step 4.k above.
   s. In the **Bootstrap script** field in **Advanced Options**, paste 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.

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, or 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.



To enable usage-based billing, you must have a Cloudera Enterprise license and a billing ID provided by Cloudera. 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, in the **Billing ID** field, enter the billing ID provided by Cloudera.

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

Cloudera Director version	Cloudera Manager version installed
Cloudera Director 2.0	Latest released version of Cloudera Manager 5.5
Cloudera Director 2.1	Latest released version of Cloudera Manager 5.7
Cloudera Director 2.2	Latest released version of Cloudera Manager 5.8
Cloudera Director 2.3	Latest released version of Cloudera Manager 5.10
Cloudera Director 2.4	Latest released version of Cloudera Manager 5.11
Cloudera Director 2.5	Latest released version of Cloudera Manager 5.12

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 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:

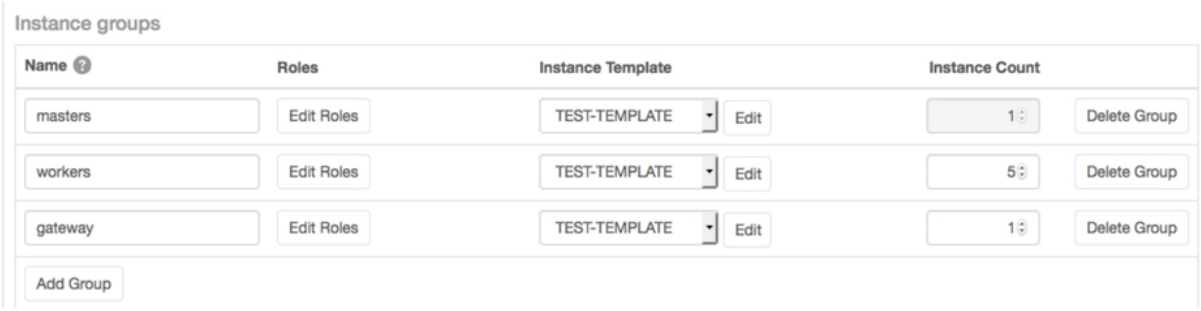
Cloudera Director version	CDH version installed
Cloudera Director 2.0	Latest released version of CDH 5.5
Cloudera Director 2.1	Latest released version of CDH 5.7
Cloudera Director 2.2	Latest released version of CDH 5.9
Cloudera Director 2.3	Latest released version of CDH 5.10
Cloudera Director 2.4	Latest released version of CDH 5.11
Cloudera Director 2.5	Latest released version of CDH 5.12

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 does not work with Cloudera Manager 5.5, but CDH 5.7 or lower works 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.



13. Click **Continue**.
14. At the confirmation prompt, click **OK** to deploy the cluster. Cloudera Director displays a status screen.

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

Terminating 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. 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, Regions, and Instances

The Cloudera Director Azure Plugin supports adding new VM images, regions, and instances by modifying configuration files. For more information see Cloudera Director Azure Plugin Config Files on the Cloudera GitHub site.

See the Cloudera Reference Architecture for Microsoft Azure Deployments for the latest supported VM images, Azure regions, and instance types.

Configuring and Deploying to Azure US Government and Azure Germany Regions

Configuring and Deploying to Azure U.S. Government Regions

For configuring and deploying to Azure U.S. Government regions, the following non-default values must be used for Cloudera Director environment configuration:

```
aadUrl: "https://login-us.microsoftonline.com/"
mgmtUrl: "https://management.core.usgovcloudapi.net/"
armUrl: "https://management.usgovcloudapi.net/"
```

Configuring and Deploying to Azure Germany Regions

The following non-default values must be used for Director Environment Configuration:

```
aadUrl: "https://login.microsoftonline.de/"
mgmtUrl: "https://management.core.cloudapi.de/"
armUrl: "https://management.microsoftazure.de/"
```

Azure Germany API endpoints use a newer CA root certificate authority called D-TRUST. For more information, see the JDK release note New DTrust certificates added to root CAs and the section Certificate Changes: New DTrust certificates added to root CAs in the Oracle Java 7 Release Notes. This newer CA root certificate authority is not currently trusted by the default JDK that is installed via the Cloudera repository, `jdk1.7.0_67-cloudera`.

In order for the plugin to work with Azure Germany, the JDK `cacerts` file must be replaced with a link to a newer version that includes the appropriate certificate with the following steps:

1. Confirm that `/etc/pki/java/cacerts` exists and contains the appropriate cert using keytool (`keytool -list -v -keystore /etc/pki/java/cacerts`). The necessary key is the one with `CN=D-TRUST Root Class 3 CA 2 2009 SHA256: y`

```
sudo mv
/usr/java/jdk1.7.0_67-cloudera/jre/lib/security/cacerts
/usr/java/jdk1.7.0_67-cloudera/jre/lib/security/cacerts.original
```

```
sudo ln -s /etc/pki/java/cacerts
/usr/java/jdk1.7.0_67-cloudera/jre/lib/security/cacerts
```

4. You may have to restart Director to get it to pickup the new trusted cert database.

> **Note:** You may have to undo the change in order for future updates to the JDK package to install properly.

The `images.conf` file must be updated to reference the proper VM image for deployment in Azure Germany. If you have not already created an `images.conf` file, you can download it as follows:

```
sudo -i
cd /var/lib/cloudera-director-plugins/azure-provider-1.3.0/etc/
wget
https://raw.githubusercontent.com/cloudera/director-scripts/master/azure-plugin-config/images.conf
```

Find and edit the following section:

```
# This is the CentOS 6.7 image published by Cloudera
cloudera-centos-6-latest {
   publisher: cloudera
   offer: cloudera-centos-6
   sku: CLOUDERA-CENTOS-6
   version: latest
}
```

...and change it to:

```
# This is the CentOS 6.7 image published by Cloudera
   cloudera-centos-6-latest {
   publisher: cloudera
   offer: cloudera-centos-os
   sku: 6_7
   version: latest
}
```

Important Notes About Cloudera Director and Azure

Azure Limits, Quotas, and Constraints

Azure limits the number of CPU cores that can be allocated in each region. For details, see Azure subscription and service limits, quotas, and constraints in the Azure documentation. If you need to increase the limit, contact Microsoft Azure support before deploying the cluster with Cloudera Director.

Not all Azure VM types are available in all Azure regions. See Products available by region on the Microsoft Azure web site to confirm that a VM type is available in a particular region. See Cloudera Reference Architecture for Microsoft Azure Deployments for the latest supported VM types.

Azure Resources Managed by Cloudera Director

The Azure plug-in for Cloudera Director creates the following resources:

- A storage account for each VM.
- A NIC for each VM.

- A public IP address for each VM, if public IP addresses are enabled.

Deploying Production Clusters

Although 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 do not 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.

See the Cloudera Reference Architecture for Microsoft Azure Deployments document for more details.

Updating the Azure Plug-in Timeout Value

Azure backend operations usually complete in a few minutes, but in rare cases they take longer, sometimes up to an hour. This can cause Cloudera Director operations such as `allocate` to fail prematurely. If this happens, you may want to increase the backend polling timeout value in the `azure-plugin.conf` file.

1. Download the latest supported `azure-plugin.conf` file from the Cloudera Director scripts repository.
2. Find the parameter `azure-backend-operation-polling-timeout-second` in the provider section.
3. Change the value to the required duration in seconds.

This procedure changes only the Azure plug-in timeout. The following Cloudera Director timeout values must also be increased in the server's `application.properties` file to be at least as large as the Azure plug-in configuration values:

- `lp.cloud.databaseServers.allocate.timeoutInMinutes`
- `lp.cloud.instances.terminate.timeoutInMinutes`

See Setting Cloudera Director Properties for information on setting configuration properties in the server's `application.properties` file.

Deletion Behavior

The deletion behavior is as follows:

- The storage account created by the plug-in 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 plug-in, it is not deleted.
- The NIC created by the plug-in is attached to the VM. Only one NIC is used per VM. Do not manually attach NICs to the VM created by the plug-in.
- Deleting the NIC also deletes the public IP attached to the NIC. This includes public IPs created by Cloudera Director as well as public IPs attached manually.

Important: Based on the deletion behaviors described, do not reuse any resources created by the Azure plug-in 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: A simple Cloudera Director configuration that creates a Cloudera Manager node and a four-node cluster (one master and three workers).
  - azure.reference.conf: A reference Cloudera Director configuration that creates an eight-node cluster (three masters and five workers) with high availability (HA) enabled.
  - azure.kerberos.conf: The same Cloudera Director configuration as azure.reference.conf, 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.

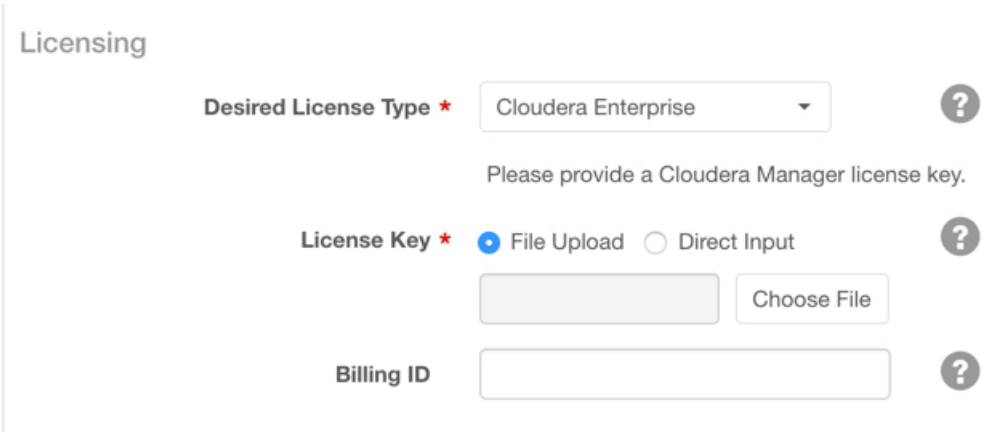
You can deploy Cloudera Manager and create clusters with usage-based billing either through the Cloudera Director server web UI or with the Cloudera Director client and the `bootstrap-remote` command, as described in this section.

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 50. 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.



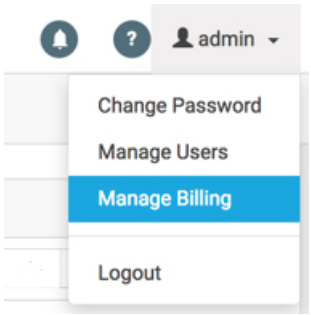
Enabling Usage-Based Billing with bootstrap-remote

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 177.

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.

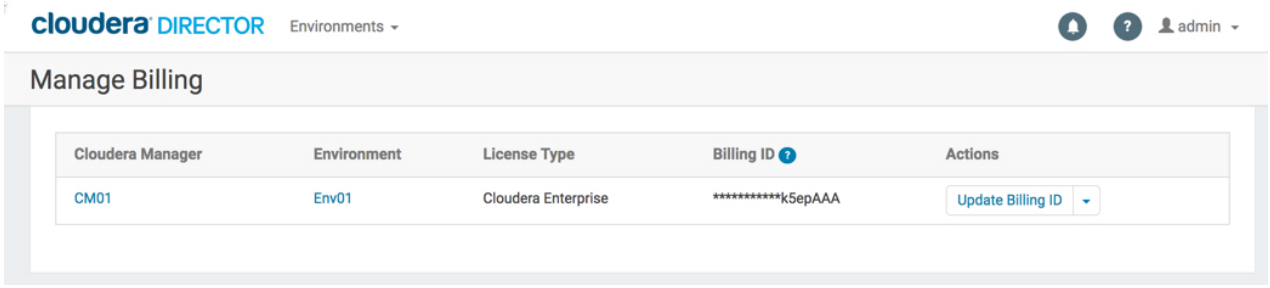
Managing Billing IDs with an Existing Deployment

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:



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.

Cloudera Director Usage Bundles

A Cloudera Director usage bundle is a JSON document representing a snapshot in time of Cloudera Manager and cluster usage. It contains three sections: a metadata section, a Cloudera Manager block, and a Cloudera Director block.

Note: All passwords and sensitive information included in Cloudera Director usage bundles are redacted.

Metadata Section

The short metadata section in a usage bundle contains:

- The version of the metadata structure
- The complete license key and billing ID for the deployment
- The creation time of the bundle
- A message ID structure, used by Cloudera's metering service for context and sequencing

Cloudera Manager Block

The Cloudera Manager block contains information queried from Cloudera Manager about itself and clusters that it manages.

Initial metadata in the block includes the metadata structure version and the host, port, and API version of Cloudera Manager itself.

Next, the block contains details about Cloudera Manager itself, as retrieved from its `/cm/deployment` API endpoint. See the Cloudera Manager REST API documentation for complete information. The data is retrieved from Cloudera Manager using its **export redacted** view, which eliminates sensitive configuration information such as passwords. This deployment data includes information about all clusters and their services, all hosts, and all management services. Some specific data items in the details are:

- Cluster, service, and role names
- Service and role configurations (redacted) and health statuses
- Cloudera Manager's internal user accounts, with redacted passwords
- Instances' Cloudera Manager host identifiers, private IP addresses, and private host names

- Instances' core counts and memory sizes

Finally, the block includes time series data for the capacity and used capacity of each filesystem associated with the Cloudera Manager instance and with every instance that is part of a cluster. The data covers the five minutes prior to the bundle's creation. See the Cloudera Manager REST API documentation for complete information on the data structures in a time series. Instance private IP addresses and host names are included in the time series data.

Cloudera Director Block

The Cloudera Director block contains information queried from Cloudera Director itself about Cloudera Manager installations and clusters that it manages. For complete information on the data structures described here, consult the Cloudera Director API documentation or explore using the API console included with Cloudera Director, at the `/api-console` URL.

Initial metadata in the block includes the metadata structure version and the host, port, and API version of Cloudera Director itself. Ensuing details begin with the version of Cloudera Director and the time when the block was created.

Next, the block includes the deployment template used to create the Cloudera Manager installation. The data retrieved from Cloudera Director here is redacted, eliminating potentially sensitive information such as external database account details and inline scripts. Some specific data items are:

- Redacted license and billing ID (which are available unredacted in the usage bundle metadata)
- External Cloudera Manager database templates, if any
- The Cloudera Manager instance template

After some deployment health and status information, details about the running deployment are included. As with the deployment template information, deployment information is redacted to eliminate potentially sensitive information such as the Cloudera Manager administrator password. Some specific data items are:

- The Cloudera Manager version and private IP address
- Details about the instance running Cloudera Manager, including its public and private IP addresses and host names, information specific to the cloud provider such as virtual network and subnet identifiers, its installed software capabilities, and its instance template
- The Cloudera Manager port and administrative username

Next, the cluster templates for each of the clusters created by Cloudera Director are listed. As with other Cloudera Director API calls, cluster template information such as inline scripts is redacted for security. Some specific data items are:

- The cluster template name and list of services deployed
- External service database templates, if any
- Virtual instance groups and associated instance templates

Finally, after some cluster status and health information, details about each bootstrapped cluster are provided. Sensitive information is left out of these query results like the rest. Some specific data items are:

- Overall cluster health and individual service health checks
- Installed software capabilities of each cluster instance

Usage Logging

Cloudera Director is capable of logging usage bundles and heartbeats as they exist immediately before submission to Cloudera's metering service. The logging is disabled by default, but it can be enabled and configured to provide visibility into precisely what Cloudera Director is sending out.

To enable usage logging in the Cloudera Director server, locate the logback.xml file used to configure its logging system. The file is normally in /usr/lib/cloudera-director-server/etc [check this]. Look for the configurations for the following loggers:

- `com.cloudera.director.metering.heartbeats`
- `com.cloudera.director.metering.bundles`

Change the **level** for each logger to **INFO** to enable usage logging. To disable usage logging, change the level back to **ERROR**. After changing the level, restart Cloudera Director so that the change takes effect.

The logging configuration writes the JSON for heartbeats and usage bundles to a dedicated log file. Those comfortable with configuring the Logback logging system can make further changes to have the information written elsewhere. Consult Logback documentation for the options available.

Usage logging increases the demand for file storage on the Cloudera Director instance. Do not enable it for long periods of time, to avoid running out of disk space.

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.

For information on the bootstrap and bootstrap-remote commands, see Commands on page 12. The configuration file follows the HOCON format. For information on HOCON, see the documentation on GitHub at HOCON (Human-Optimized Config Object Notation).

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 or *cloud_provider*.reference.conf file to *your_filename*.conf.

- For simple cluster configuration, use *cloud-provider*.simple.conf.
- For advanced cluster configuration, use *cloud_provider*.reference.conf.

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

Open your copy of the configuration with a text editor to customize the configuration settings.

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 the API to Import a Configuration File

Beginning with release 2.0 of Cloudera Director, you can import the contents of a configuration file into Cloudera Director with the API using the `/api/<api_version>/import` endpoint:

```
/api/v9/import?clusterName=someClusterName&deploymentName=someDeploymentName&environmentName=someEnvironmentName
```

In this way, you can POST the contents of a configuration file to Cloudera Director without using the Cloudera Director CLI. The values clusterName, deploymentName, and environmentName are all optional. If they are present, they override the values in the configuration file.

Using Spot Instances

To help you manage your cloud resource costs, Cloudera supports Spot instances. Spot instances are Amazon EC2 instances that you can bid on. They run just like On-Demand instances, except that they are not provisioned until the instance price falls below your bid price. Amazon will terminate the instances when the instance price exceeds your bid price. Spot instances allow you to add capacity to your clusters at a low price.

Cloudera Director supports Spot instances for worker roles on the following services:

- YARN
- Hive on Spark
- Hive on MapReduce
- Spark

For all other components and for master roles, Cloudera recommends using on-demand instances.

Note: Amazon EC2 does not currently support Spot instances on Red Hat Enterprise Linux (RHEL). Spot instances must be run on supported Centos AMIs.

Although Spot instances are supported with Cloudera Director 2.0 and higher, Cloudera recommends using Cloudera Director 2.5 and higher together with Cloudera Manager and CDH 5.12 and higher for Spot instances. Beginning with these versions, Cloudera has improved cluster resiliency to support Spot instances better. Now, if Amazon terminates a Spot instance during a bootstrap or update operation, Cloudera Director will shrink the terminated instance out of the cluster, as long as the affected instance group still meets the minimum count requirement. For this reason, Cloudera recommends specifying a minimum count of zero for instance groups that use Spot instances. (See Best Practices for Using Spot Instances on page 104 below.)

Note: With versions of Cloudera Director earlier than 2.5 and versions of Cloudera Manager and CDH earlier than 5.12, a cluster will fail if Amazon terminates a Spot instance during a bootstrap or update operation. If a bootstrap operation fails, *it cannot be recovered*, and a new cluster must be launched in its place. If an update operation fails, it may be possible to recover cluster functionality, but *only with the assistance of Cloudera Support*.

Note: Because Spot instances are terminated by AWS if the Spot price you bid becomes lower than the current Spot price, you should not use Spot instances for master nodes or for data storage. Use Spot instances only for instances where the loss of the instance can be tolerated, such as compute nodes or other worker nodes.

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 use Cloudera Manager to 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.

You cannot do the following:

- Change the bid price for spot instances.

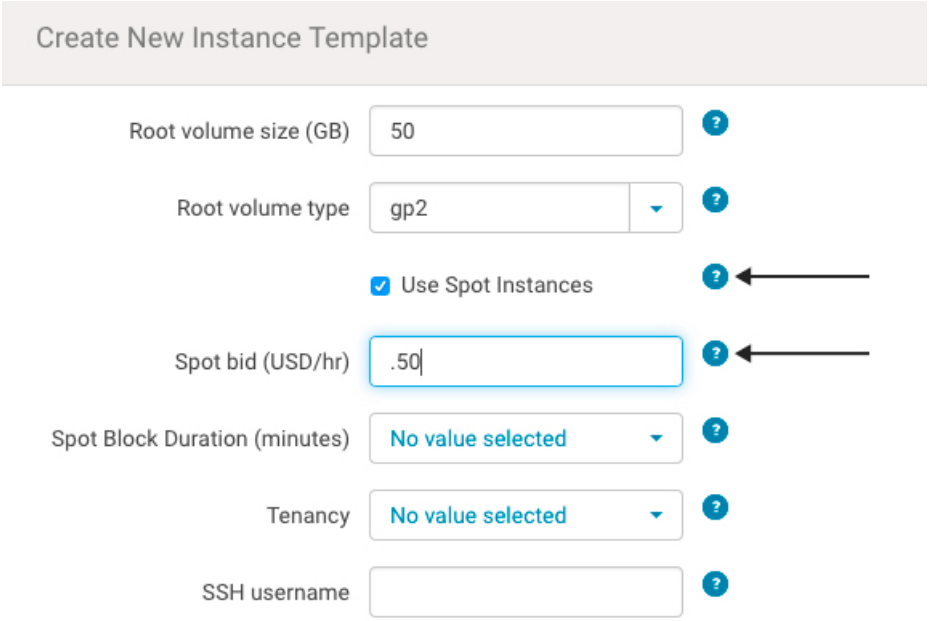
> **Note:** Although you cannot change the bid price for a spot instance, you can work around this restriction by adding a new, identical virtual instance group with the desired bid price.

- Delete all instances without adding more, due to the minimum instance count requirement.

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

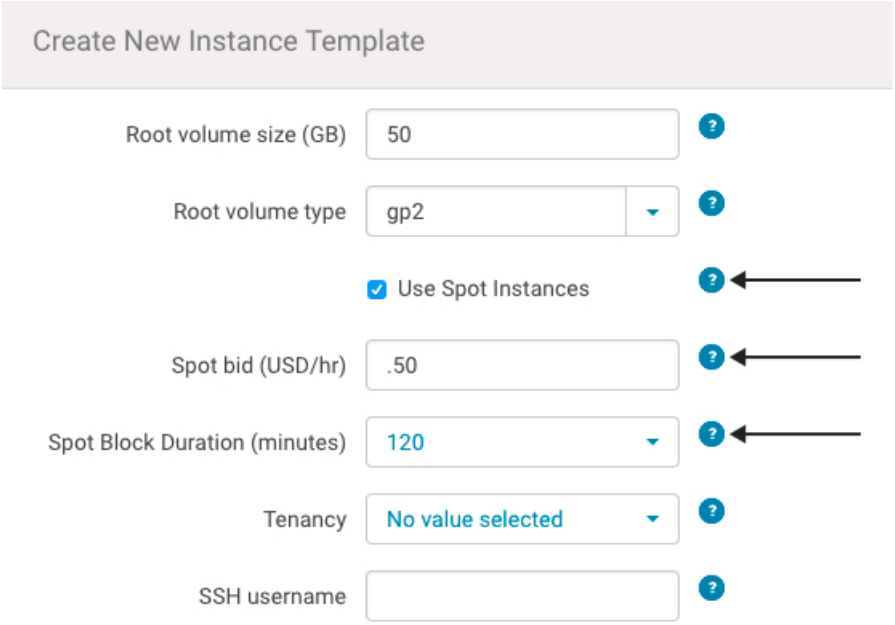
To specify Spot instances, create a new instance template and use this template for your instance group. In the **Advanced Options** section of the **Create New Instance Template** wizard, check the **Use Spot Instances** checkbox and enter a value in the **Spot bid** field.



Spot Blocks: Specifying a Duration for Spot Instances

The AWS Spot block feature enables you to specify a fixed duration ranging from one to six hours for Spot instances. Spot instances with a predefined duration use a fixed hourly price that remains in effect for the Spot instance while it runs. The price for instances with a Spot block will not be as low as that for ordinary Spot instances, but will be lower than that of on-demand instances, and Spot block instances are guaranteed to run for the specified duration, after which they are terminated.

To configure an instance template for Spot block instances, check the **Use Spot Instances** checkbox, enter a value in the **Spot bid** field, and choose a value from one to six hours (in intervals of 60 minutes) in the **Spot Block Duration** dropdown.



See Specifying a Duration for Your Spot Instances in the AWS documentation for more information about Spot blocks.

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.

Additional best practices for using spot instances can be found in Spot Instance Interruptions in the AWS documentation.

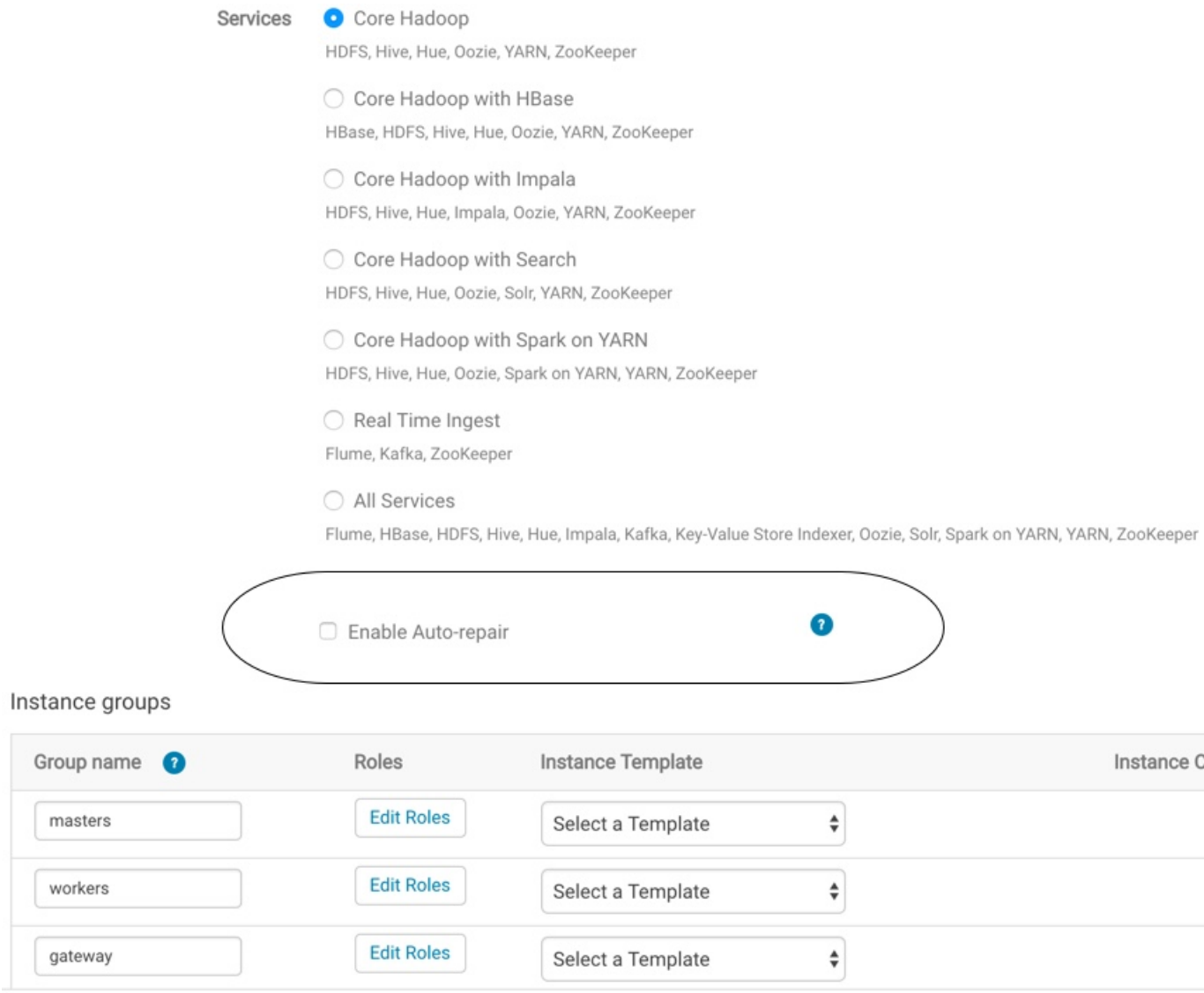
Auto-Repair for Failed or Terminated Instances

Cloudera Director 2.5 and higher with Cloudera Manager 5.12 or higher includes an auto-repair feature that allocates new cluster instances to replace failed or terminated instances. For clusters running on AWS, this includes Spot instances that were terminated by Amazon because your bid price became lower than Amazon's current Spot instance price. Auto-repair checks to ensure that the number of instances in your cluster matches the number that was specified in the cluster template.

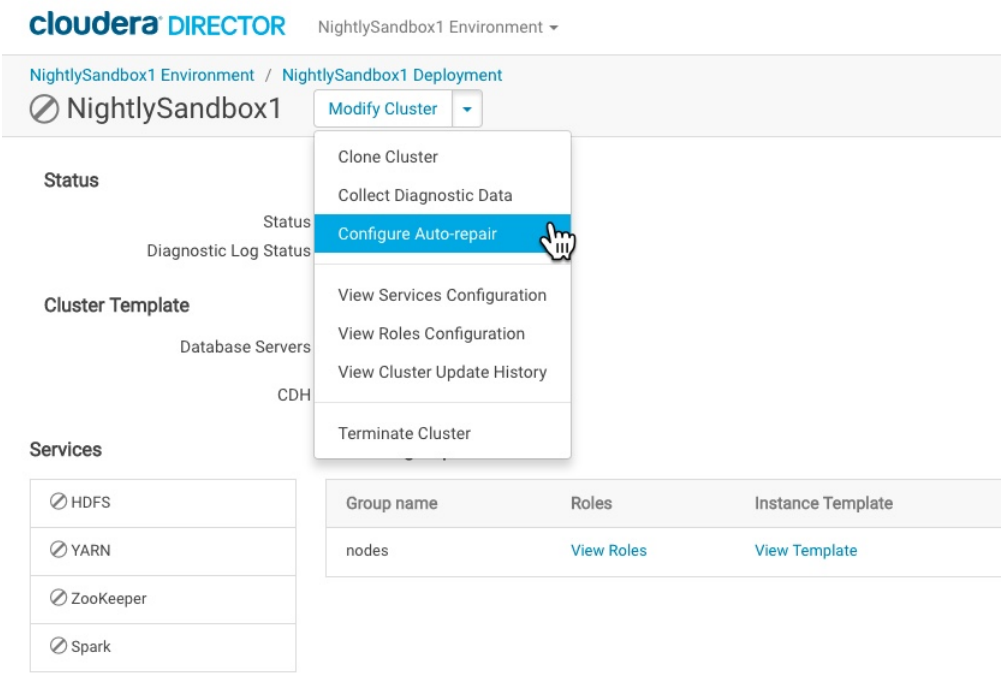
> **Note:** If auto-repair is enabled on a cluster that uses Elastic Block Storage (EBS), you must disable auto-repair for the cluster before stopping either the cluster or the Cloudera Manager instance that manages it. For information about stopping and starting clusters that use EBS volumes, see Using EBS Volumes for Cloudera Manager and CDH on page 141.

By default, auto-repair is not enabled. You can enable auto-repair in one of the following three ways:

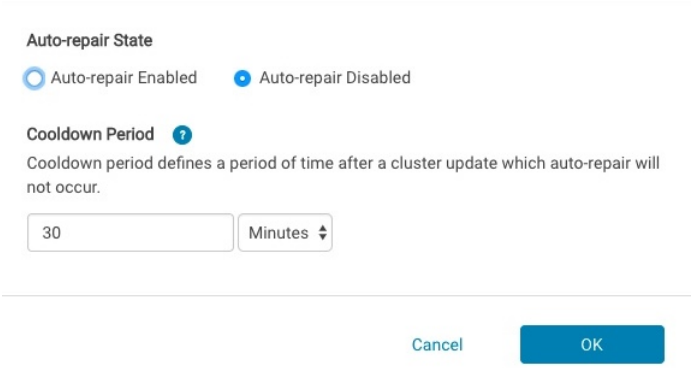
- During the installation process while configuring settings for the cluster:



- In the **Modify Cluster** dropdown for an existing cluster:



Choosing **Configure Auto-repair** in the **Modify Cluster** drop-down opens a dialog for enabling or disabling the auto-repair feature or configuring the cooldown period, which is the time that elapses between Cloudera Director's attempts to allocate new cluster instances:



- If you are launching a cluster with the bootstrap-remote CLI command and a configuration file, you can enable auto-repair by setting autoRepairEnabled to true in the administrationSettings section of the configuration file:

```
administrationSettings {
   # If enabled, Director will attempt to automatically repair
   # clusters whose instances have been terminated in the cloud provider.

   # autoRepairEnabled: false
   # autoRepairCooldownPeriodInSeconds: 1800
}
```

Keep in mind the following facts about the auto-repair feature:

- Auto-repair is only available with Cloudera Director 2.5 and higher running with Cloudera Manager 5.12 and higher.
- Auto-repair only functions with instances that do *not* contain master roles.
- Before stopping a cluster using the Elastic Block Storage (EBS) start/stop feature, you must disable auto-repair if it is enabled.
- Auto-repair is disabled by default, and can be enabled (1) when creating a cluster in the web UI, (2) on the web UI page for an existing cluster, or (3) in the configuration file when launching a cluster with bootstrap-remote.

- Auto-repair is only available for clusters managed with Cloudera Director server; the feature is not available for clusters running in the standalone client version of Cloudera Director.
- Auto-repair will attempt to allocate the same type of instance that is missing from the cluster. So for an on-demand instance, another on-demand instance with identical specifications will be allocated. For a Spot instance, another Spot instance with the same bid price will be allocated.
- The auto-repair cooldown period ensures that, even if auto-repair is enabled and your cluster has fewer instances than specified in the cluster template, Cloudera Director will not continuously attempt to repair the cluster, and you will therefore have frequent intervals when you can interact with the cluster to perform other tasks. The cooldown period is configurable in the web UI or the configuration file.

Using Custom Repositories with Cloudera Manager and CDH

You can create a custom repository to use when deploying Cloudera Manager and CDH, for example, if your organization restricts internet access to the Cloudera repositories, or if you wish to use custom AMIs preloaded with Cloudera Manager packages and CDH parcels.

Creating a Custom Repository

See Creating and Using a Package Repository for Cloudera Manager in the Cloudera Manager documentation for instructions and a link for downloading the Cloudera Manager repository for various operating systems. Following these instructions will ensure that your repository is created with the required directory structure. Be sure to choose an operating system that is supported by Cloudera Director.

Creating a Cloudera Manager and CDH AMI

For clusters running on AWS EC2 instances, you can reduce 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 the README.md file on the Cloudera GitHub site.

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 Microsoft Azure or 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 SSH access on port 22 from the IP address of the Cloudera Director host.
  - **For AWS:** If you are creating the cluster with the UI, set **Associate public IP addresses** to **true** in the Environment for Cloudera Manager and the cluster. If you are creating the cluster with the CLI, set the **associatePublicIpAddresses** to **true** in the configuration file.
  - **For Microsoft Azure:** If you are creating the cluster with the UI, set **Public IP** to **Yes** in the instance template for Cloudera Manager and the cluster. If you are creating the cluster with the CLI, set **publicIP** to **Yes** in the configuration file.
- 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 7180) 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.

Using a New AWS Region in Cloudera Director

Cloudera Director's AWS support, embodied in a plugin, ships with a predefined, known set of AWS regions. Cloudera adds support for additional regions when possible in new Cloudera Director releases. But, because you may want to use a new region before it has been added in a new release, Cloudera Director makes it possible to add the new region yourself.

> **Note:** Examples here use the region code `xy-east-1` as an example of a new region. Use the code for the region you want to use instead.

For more information about AWS regions, see Regions and Availability Zones in the AWS documentation.

Entering the Region Code

When using its web interface, Cloudera Director asks you which region to use when you define a new AWS environment. You can select the region for EC2, where instances hosting Cloudera Manager and cluster components run, and for RDS, where an external database server can house databases for Cloudera Manager and services like Hive and Oozie.

The region selection widgets are ordinary drop-down menus, but the menus are also editable. To use a region that isn't listed, just type in its region code.

When you use Cloudera Director's configuration file support for defining new deployments and clusters, you don't have any widgets. Simply supply the region code for EC2 and RDS in the expected locations.

- EC2 region: in the `provider` section, as the `region` field
- RDS region: in the `provider` section, as the `rdsRegion` field. If the region is not specified, it defaults to the EC2 region

Region Endpoints

In most cases, Cloudera Director can figure out the AWS endpoints for the different services in a region, so just naming the new region is enough to get things moving. If you receive errors that an AWS service could not be reached, you may need to specify some endpoints, as described below for RDS, IAM, and KMS.

For general information about region endpoints in AWS, see AWS Regions and Endpoints in the AWS documentation.

RDS

If you plan on using RDS, you must supply the RDS endpoint for your chosen region. There are two ways to do this.

- Using the web UI interface, specify the endpoint URL directly when you define your environment. In the web interface, expand the **Advanced Options** section under **RDS (Relational Database Service)** and enter the endpoint URL for **RDS region endpoint**. In a configuration file, give the URL as the value for the `rdsRegionEndpoint` field in the `provider` section. Here is what an endpoint URL looks like:

```
rdsRegionEndpoint: https://rds.xy-east-1.amazonaws.com
```

- Rather than specifying the RDS endpoint URL with each environment you create, you can supply it in a configuration file that is read by Cloudera Director's AWS plugin, so it will be used for all environments created with that instance of Cloudera Director. The configuration file is named `rds.endpoints.properties` and, by default, resides in the directory `/var/lib/cloudera-director-plugins/aws-provider-`*`version`*`/etc/`. The version number for the `aws-provider` part of the path changes with most Cloudera Director releases, as the plugin changes version. For example, `aws-provider-1.4.1` matches with Cloudera Director 2.4. So the path and file name with Cloudera Director 2.4 would be as follows:

```
/var/lib/cloudera-director-plugins/aws-provider-1.4.1/etc/rds.endpoints.properties
```

  Cloudera Director ships with an example of the file that you can use as a template: `rds.endpoints.properties.example`. Copy this file to a new `rds.endpoints.properties` file in that directory, and add a line for the RDS endpoint URL, for example:

```
xy-east-1=https://rds.xy-east-1.amazonaws.com
```

After adding a new endpoint, restart Cloudera Director if it is running.

IAM

The IAM service is normally accessed using a single, global endpoint that works across all AWS regions. Some regions, however, have their own IAM endpoint. If you are using such a region, supply its custom IAM endpoint. When using the web interface, expand the **Advanced Options** section under **EC2 (Elastic Compute Cloud)** on the environment page, and enter the endpoint URL for **IAM endpoint**. In a configuration file, specify it in the field `iamEndpoint` in the `provider` section.

```
iamEndpoint: https://iam.xy-east-1.amazonaws.com
```

KMS

Cloudera Director normally computes the expected KMS endpoint for your chosen region. If that process fails, then you can provide the endpoint URL yourself. In the web interface, expand the **Advanced Options** section under **EC2 (Elastic Compute Cloud)** on the environment page, and enter the endpoint URL for **KMS region endpoint**. In a configuration file, specify it in the field `kmsEndpoint` in the `provider` section.

```
kmsEndpoint: https://kms.xy-east-1.amazonaws.com
```

Other Considerations

A new AWS region usually does not support the full range of services and features that are available in older, established regions. It's important to understand what services and features your chosen region lacks, so that you do not request them through Cloudera Director. Cloudera Director does not retain knowledge on which regions have which services available.

Here are some examples of items that can work in older regions but not fully, or at all, in newer ones.

- AMIs - common "stock" AMIs may not exist for new regions
- instance types - deprecated instance types are often left out of new regions

- dedicated instances (tenancy)
- Spot blocks
- RDS instance encryption

Using Products outside CDH with Cloudera Director

Products packaged in parcels separate from the CDH parcel may be included in clusters that are bootstrapped with Cloudera Director. Create a configuration file that includes the desired products (see The Cloudera Director Configuration File on page 101 for more information), and then use the Cloudera Director CLI to bootstrap a cluster with the services.

Custom Service Descriptors

A custom service descriptor (CSD) is a file that describes a product for use with Cloudera Manager. When a product is accompanied with a CSD, Cloudera Manager can support configuration, distribution, and monitoring of that product. See CSD Overview for an overview of CSDs.

Most products available via parcel are accompanied by a CSD. Starting in Director 2.4, you can point to CSDs that Cloudera Director should download and install into Cloudera Manager during the bootstrap process. Once a CSD is installed, its corresponding product distributed in a parcel can then be installed properly in clusters managed by Cloudera Manager.

To install a CSD into Cloudera Manager, provide a URL for it in the deployment template, in the `csds` section.

```
```
cloudera-manager {
 ...
 csds: [
    "http://archive.cloudera.com/spark2/csd/SPARK2_ON_YARN-2.0.0.cloudera2.jar",
    "http://archive.cloudera.com/kudu/csd/KUDU-5.10.0.jar",
  ]
}
```
```

During deployment bootstrap, Cloudera Director will download each CSD from its URL and install it into the correct location for Cloudera Manager to recognize it.

Custom CSD Installation Directory

You may define a non-default location for CSD installation using the `csd_repo_path` Cloudera Manager server configuration property.

```
```
...
cloudera-manager {
 ...
 configs {
 csd_repo_path: /custom/path/to/csds
 ...
 }
}
```
```

When this configuration property is specified in a deployment template, Cloudera Director installs CSD files into the custom location instead of the default location.

CSD Support Before Cloudera Director 2.4

The ability to define CSDs for installation in deployment templates is a new feature for Cloudera Director 2.4. For earlier versions of Cloudera Director, use a different mechanism to install CSD files for Cloudera Manager. Here are some options:

- Use a deployment post-creation script to download CSDs to the newly allocated Cloudera Manager instance. See Deployment Post-creation Scripts for more information.

- Use an image for the Cloudera Manager instance which already includes CSDs.

Installing CSDs after Deployment Bootstrap

You may install additional CSDs into Cloudera Manager after Cloudera Director has bootstrapped it. Cloudera Manager must be restarted for it to recognize the new CSDs. Future bootstraps of new clusters that use the updated Cloudera Manager installation may include the products corresponding to the CSDs, as if Cloudera Director itself had originally installed them.

Using Kudu with Cloudera Director

The Kudu service is distributed in its own parcel and is not part of CDH. To add Kudu to a cluster bootstrapped by Cloudera Director, perform the following steps.

1. List "KUDU" as a product in the cluster template, providing its version number.
2. Include the URL for a Kudu parcel repository in the list of parcel repositories for the cluster template. Be sure to also include the URL for the CDH parcel repository, even if it is the default repository that Cloudera Director uses when no parcel repositories are listed.
3. Manually assign roles for Kudu, as well as other products, to instances in the cluster template.
4. Provide required Kudu configuration properties, such as the paths for the data and write-ahead log files.

```
```
cluster {
  products {
    CDH: 5.11.0,
    KUDU: 1.2.0
  }

  parcelRepositories: ["http://archive.cloudera.com/cdh5/parcels/5.11.0/",
                       "http://archive.cloudera.com/kudu/parcels/5.11.0/"]

  services: [HDFS, YARN, KUDU]

  masters {
    count: 1
    instance: {
      type: m4.xlarge
      image: ami-12345678
    }

    roles {
      HDFS: [NAMENODE, SECONDARYNAMENODE]
      YARN: [RESOURCEMANAGER, JOBHISTORY]
      KUDU: [KUDU_MASTER]
    }

    configs {
      KUDU {
        KUDU_MASTER {
          fs_wal_dir: "/data0/kudu/masterwal"
          fs_data_dirs: "/data1/kudu/master"
        }
      }
    }
 }
  workers {
    count: 3
    minCount: 3
    instance: {
      type: m4.xlarge
      image: ami-12345678
    }

    roles {
      HDFS: [DATANODE]
      YARN: [NODEMANAGER]
      KUDU: [KUDU_TSERVER]
    }
```
```

```
    configs {
      KUDU {
        KUDU_TSERVER {
          fs_wal_dir: "/data0/kudu/tabletwal"
          fs_data_dirs: "/data1/kudu/tablet"
        }
      }
    }
  }
}
```

The Kudu configurations above are examples only. Check the Kudu documentation for the complete set of required configurations. Also note that valid paths for files depend on the file systems that are established in cluster instances, which depend on the underlying operating system images.

The CSD for Kudu is included with Cloudera Manager starting with version 5.11. If you are using that version or later of Cloudera Manager, you do not need to list the Kudu CSD in the `csds` section of the deployment template. Otherwise, do include the CSD.

```
cloudera-manager {
  ...
  # Kudu CSD is not included with Cloudera Manager 5.10
  csds: [
     "http://archive.cloudera.com/kudu/csd/KUDU-5.10.0.jar"
   ]
}
```

To learn more about using Kudu alongside CDH, see Installing Kudu in the Cloudera Enterprise documentation.

Using Spark 2 with Cloudera Director

The Spark 2 service is distributed in its own parcel and is not part of CDH. (CDH includes Spark 1, but Spark 2 may be installed alongside Spark 1 in the same cluster.)

Note: Spark 2.2 requires JDK 8 and Python 2.7 or higher. For information on using Java 8 with Cloudera Director, see Deploying a Java 8 Cluster.

To add Spark 2 to a cluster bootstrapped by Cloudera Director, perform the following steps.

1. List "SPARK2" as a product in the cluster template, providing its version number.
2. Include the URL for a Spark 2 parcel repository in the list of parcel repositories for the cluster template. Be sure to also include the URL for the CDH parcel repository, even if it is the default repository that Cloudera Director uses when no parcel repositories are listed.
3. Manually assign roles for Spark 2, as well as other services, to instances in the cluster template.
4. Provide the URL for the corresponding Spark 2 CSD in the list of CSDs in the deployment template.

```
cloudera-manager {
  ...
  csds: [
     "http://archive.cloudera.com/spark2/csd/SPARK2_ON_YARN-2.0.0.cloudera2.jar"
     "http://archive.cloudera.com/kudu/csd/KUDU-5.10.0.jar",
  ]
}

cluster {
  products {
    CDH: 5.11.0,
    SPARK2: 2.0.0.cloudera2
  }
  parcelRepositories: ["http://archive.cloudera.com/cdh5/parcels/5.11.0/",
```

```
                        "http://archive.cloudera.com/spark2/parcels/2.0.0.cloudera2/"]

  services: [HDFS, YARN, SPARK2_ON_YARN]

  masters {
    count: 1
    instance: {
      type: m4.xlarge
      image: ami-12345678
    }

    roles {
      HDFS: [NAMENODE, SECONDARYNAMENODE]
      YARN: [RESOURCEMANAGER, JOBHISTORY]
      SPARK2_ON_YARN: [SPARK2_YARN_HISTORY_SERVER]
    }
  }

  workers {
    count: 3
    minCount: 3
    instance: {
      type: m4.xlarge
      image: ami-12345678
    }
    roles {
      HDFS: [DATANODE]
      YARN: [NODEMANAGER]
    }
  }
}
```

To learn more about using Spark 2 alongside CDH, see Cloudera Distribution of Apache Spark 2 Overview.

Using Third-Party Products with Cloudera Director

Some products created by Cloudera partners are packaged as separate parcels, with or without accompanying CSDs. These can be included in clusters bootstrapped by Cloudera Director.

Required Information

The following information is needed to successfully include a third-party product in a Cloudera Director configuration file. If you cannot determine this information yourself, consult the partner documentation for the product.

- The URL for the directory containing the desired product parcel. This is the parcel repository URL that must be included in the Cloudera Director cluster template. Parcels for different operating systems may be co-located in one repository. It is necessary that this directory also contain a standard manifest.json file that is interpreted by Cloudera Manager to select the appropriate parcel for a cluster.
- The name of the product, which is usually the initial part of the parcel file name.
- The URL for the product's corresponding CSD, if one is provided. Some products do not require CSDs.
- The names of the services that comprise the product, and the roles that comprise each service. These are listed in the SDL file that is part of the CSD.
- Any required configuration properties for services or roles.

Additions to a Cloudera Director Configuration File

To add a third-party product to a cluster bootstrapped by Cloudera Director, perform the following steps:

1. List the name of the parcel product in the cluster template, providing its version number.
2. Include the URL for the product's parcel repository in the list of parcel repositories for the cluster template. Be sure to also include the URL for the CDH parcel repository, even if it is the default repository that Cloudera Director uses when no parcel repositories are listed.
3. Manually assign roles for each desired service in the product, as well as other services like those included in CDH, to instances in the cluster template.

4. Provide the URL for the corresponding product CSD, if provided, in the list of CSDs in the deployment template.
5. Provide any required configuration properties for the third-party services or roles.

Validation Warnings

Cloudera Director does not validate the services and roles that are part of a third-party product. It is normal for Cloudera Director to report validation warnings for them, for example, for a service named "EXTRA_SERVICE":

```
```
* Unknown service type: EXTRA_SERVICE. Skipping role type validation.
```
```

These warnings are normal, and Cloudera Director will proceed with the bootstrap process. Check Cloudera Manager for details on any product installation or configuration errors.

Missing manifest.json File

Cloudera Director may fail with a validation error if the parcel repository URL for a third-party product is missing a manifest file.

```
```
* Unsuccessful response from URL: http://archive.example.com/product/1.2.3/manifest.json,
 http code: 404
```
```

Verify that the correct parcel repository URL was provided in the cluster template; it should be the directory that contains the desired parcel. If the directory is correct, but no `manifest.json` file is present, then you can self-host the parcel(s) for the product and generate your own `manifest.json` file. See The parcel repository format on the Cloudera GitHub site for the specification of a parcel repository, including for the `manifest.json` file.

Seeking Help

Third-party products distributed via parcel for use with Cloudera Manager are supported by Cloudera partners. This support includes providing the necessary information for installing their products using CSDs and parcels, describing required service and role configurations, and general documentation on using the products. Please contact Cloudera partners' support organizations for help with any of these items.

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 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
        bootstrapScriptsPaths: ["/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 a different version of CentOS/RHEL 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 following topics in the AWS documentation:

- For an introduction to IAM, see AWS Identity and Access Management User Guide.
- For instructions on how to create an IAM role, see Creating a Role to Delegate Permissions to an AWS Service.
- For information on using IAM policies to manage access to Amazon RDS resources, see Using Identity-Based Policies (IAM Policies) for Amazon RDS.
- For information on constructing Amazon Resource Names (ARNs) for Amazon RDS resources, see Working with Amazon Resource Names (ARNs) in Amazon RDS.

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
  - DescribeNetworkAcls
  - DescribeSubnets
  - DescribeInstanceAttribute
  - RunInstances

  - TerminateInstances

- To use EBS volumes, the following additional EC2 permissions are required:
  - CreateVolume
  - DescribeVolumes
  - AttachVolume
  - DeleteVolume
  - ModifyInstanceAttribute
- To use spot instances, the following additional EC2 permissions are required:
  - RequestSpotInstances
  - CancelSpotInstanceRequests
  - DescribeSpotInstanceRequests
- When working with EBS volumes, in order to use a custom key stored in KMS for EBS encryption, Cloudera Director also requires the following KMS permission:
  - DescribeKey
- To use the importKeyPairIfMissing property, Cloudera Director requires the following EC2 permission:
  - ImportKeyPair
- 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
  - DescribeDBSubnetGroups

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.

```
{
    "Version": "2012-10-17",
    "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:DescribeNetworkAcls",
                "ec2:DescribeSubnets",
```

```
                "ec2:DescribeInstanceAttribute",
                "ec2:RunInstances",
                "ec2:TerminateInstances",
                "ec2:CreateVolume",
                "ec2:DescribeVolumes",
                "ec2:AttachVolume",
                "ec2:DeleteVolume",
                "ec2:ModifyInstanceAttribute",
                "ec2:ImportKeyPair",
                "ec2:RequestSpotInstances",
                "ec2:CancelSpotInstanceRequests",
                "ec2:DescribeSpotInstanceRequests"
            ],
            "Resource": "*"
        },
        {
            "Sid": "directorKms",
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
            "Sid": "directorIam",
            "Effect": "Allow",
            "Action": [
                "iam:GetInstanceProfile",
                "iam:PassRole"
            ],
            "Resource": "*"
        },
        {
            "Sid": "directorRds",
            "Effect": "Allow",
            "Action": [
                "rds:CreateDBInstance",
                "rds:DeleteDBInstance",
                "rds:DescribeDBInstances",
                "rds:DescribeDBEngineVersions"
            ],
            "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 143.

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 119 to verify that your MySQL configuration meets the requirements for Cloudera Director.
- The `datadir` directory (`/var/lib/mysql` by default) must be located on a partition that has sufficient free space.

1. Install the MySQL database.

OS	Command
RHEL	`$ sudo yum install mysql-server`

SLES	`$ sudo zypper install mysql`
	`$ sudo zypper install libmysqlclient_r15`
	Note: Some SLES systems encounter errors with the `zypper install` command. For more information, see the Novell Knowledgebase topic, error running chkconfig.

Ubuntu and Debian	`$ sudo apt-get install mysql-server`

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

Configuring and Starting the MySQL Server

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

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

3. Move old InnoDB log files `/var/lib/mysql/ib_logfile0` and `/var/lib/mysql/ib_logfile1` from `/var/lib/mysql/` to a backup location.
4. Determine the location of the option file, `my.cnf`, and update it as follows::
   - To prevent deadlocks, set the isolation level to read committed.
   - Configure MySQL to use the `InnoDB` engine, rather than `MyISAM`. (The default storage engine for MySQL is `MyISAM`.) To check which engine your tables are using, run the following command from the MySQL shell:

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

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

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

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

Following is a typical option file:

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

key_buffer_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	`$ 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`

SLES	`$ sudo chkconfig --add mysql`
Ubuntu and Debian	`$ sudo chkconfig mysql on` **Note:** `chkconfig` 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.

7. Start the MySQL server:

OS	Command
RHEL	`$ sudo service mysqld start`
SLES, Ubuntu, and Debian	`$ sudo service mysql start`

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	1. Download the MySQL JDBC driver from the Download Connector/J page of the MySQL web site.
	2. Extract the JDBC driver JAR file from the downloaded file. For example:
	`tar zxvf mysql-connector-java-version.tar.gz`
	3. Add the JDBC driver, renamed, to the relevant server. For example:
	`$ sudo cp mysql-connector-java-version/mysql-connector-java-version-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-version/mysql-connector-java-version-bin.jar /usr/share/java/mysql-connector-java.jar`
	Note: Do not use the `yum install` command to install the MySQL connector package, because it installs the openJDK, and then uses the Linux `alternatives` command to set the system JDK to be the openJDK.

SLES	`$ sudo zypper install mysql-connector-java`
Ubuntu or Debian	`$ sudo apt-get install libmysql-java`

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:
```

Migrating Cloudera Director Server from H2 to MySQL

If you are already running Cloudera Director with an embedded H2 database, you can migrate to a MySQL database by following the steps in this section. The SQL dialect that H2 uses is not compatible with MySQL, so this process requires the use of SQuirreL SQL, which translates the SQL dialect of H2 into that of MySQL. Cloudera Director will be used to create the schema in the MySQL database because SQuirrel SQL's translation will not create the database schema exactly as Cloudera Director requires.

Note: Cloudera strongly recommends using MySQL or MariaDB for production deployments of Cloudera Director, instead of H2.

Step 1: Preparation of databases

1. Stop Cloudera Director and back up the H2 database file, `state.h2.db`. The H2 database file is located at `/var/lib/cloudera-director-server/state.h2.db`.
2. Create a user on the MySQL server for Cloudera Director:

```
CREATE USER director IDENTIFIED BY password;
```

3. Create a database on the MySQL server for exporting the data from H2:

```
CREATE DATABASE directorexport CHARACTER SET utf8;
GRANT ALL PRIVILEGES ON directorexport.* TO 'director'@'%';
```

4. Create a database on the MySQL server for use by Cloudera Director:

```
CREATE DATABASE director CHARACTER SET utf8;
GRANT ALL PRIVILEGES ON director.* TO 'director'@'%';
```

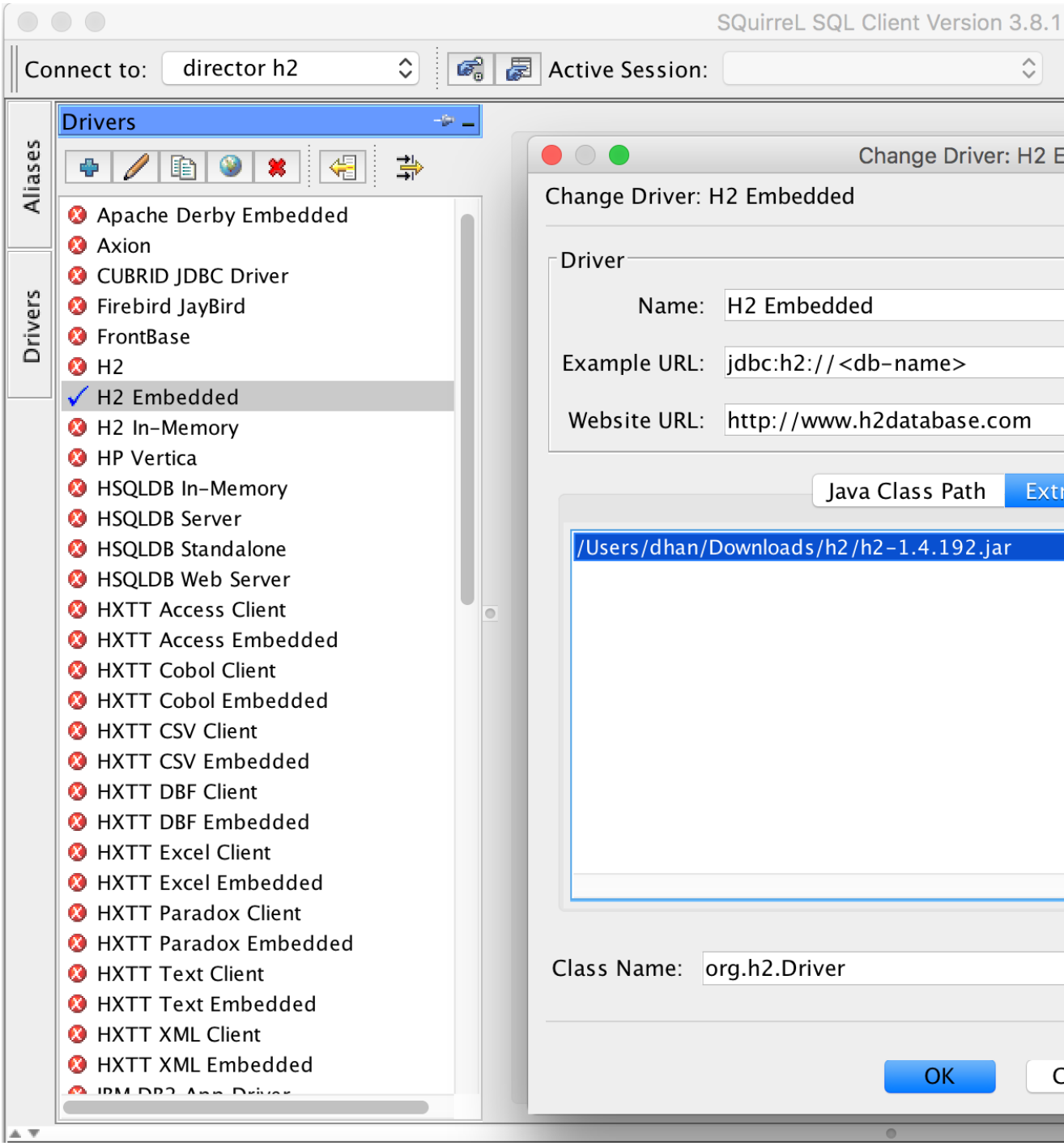
Step 2: Export data from H2 database

1. Install SQuirreL SQL.

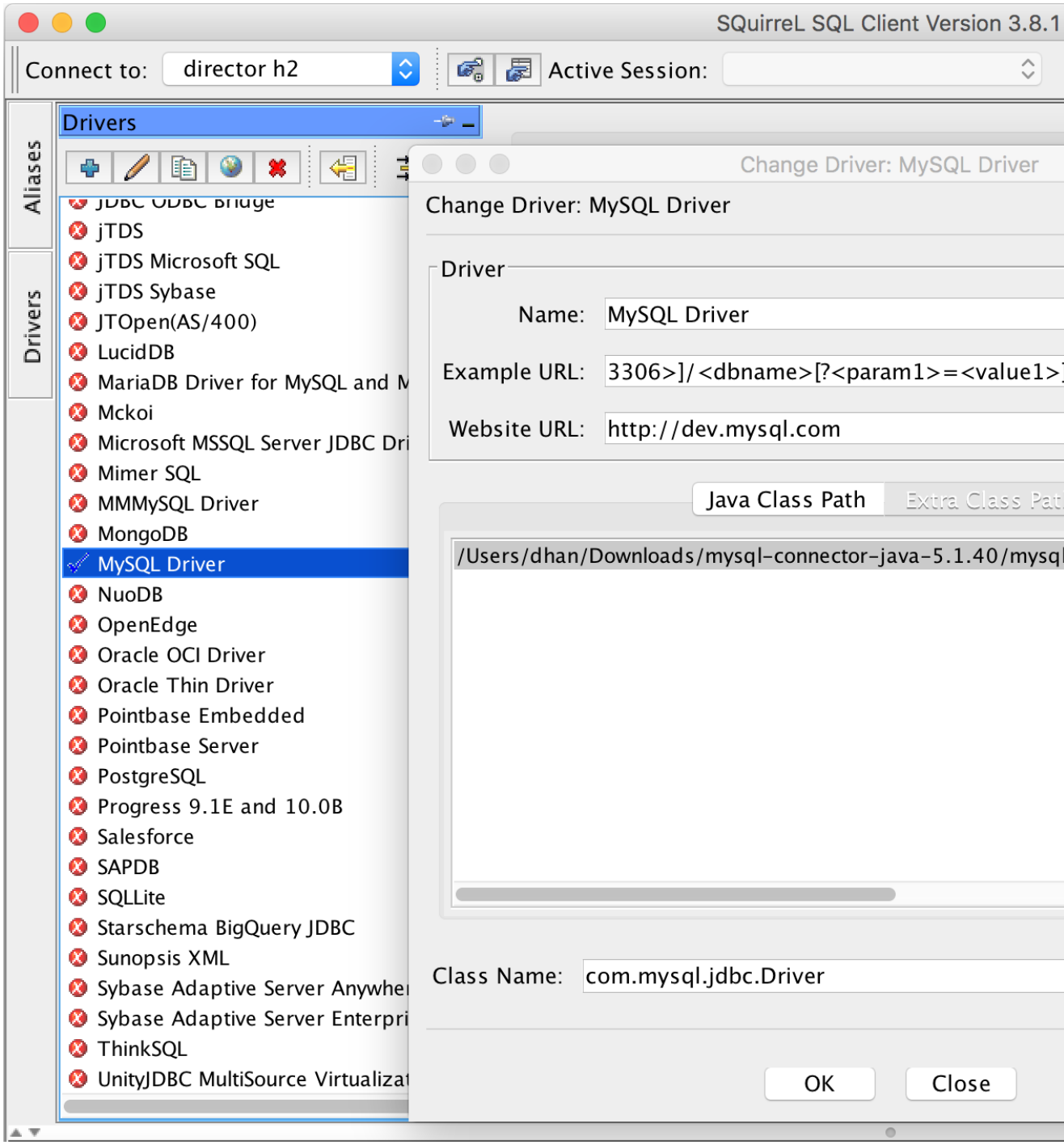
Note: H2, MySQL, and SQuirreL SQL can be run on different nodes.

   a. Download SQuirreL SQL from the SQuirreL SQL web site.
   b. Install SQuirreL SQL with the H2 and MySQL database plugins enabled.

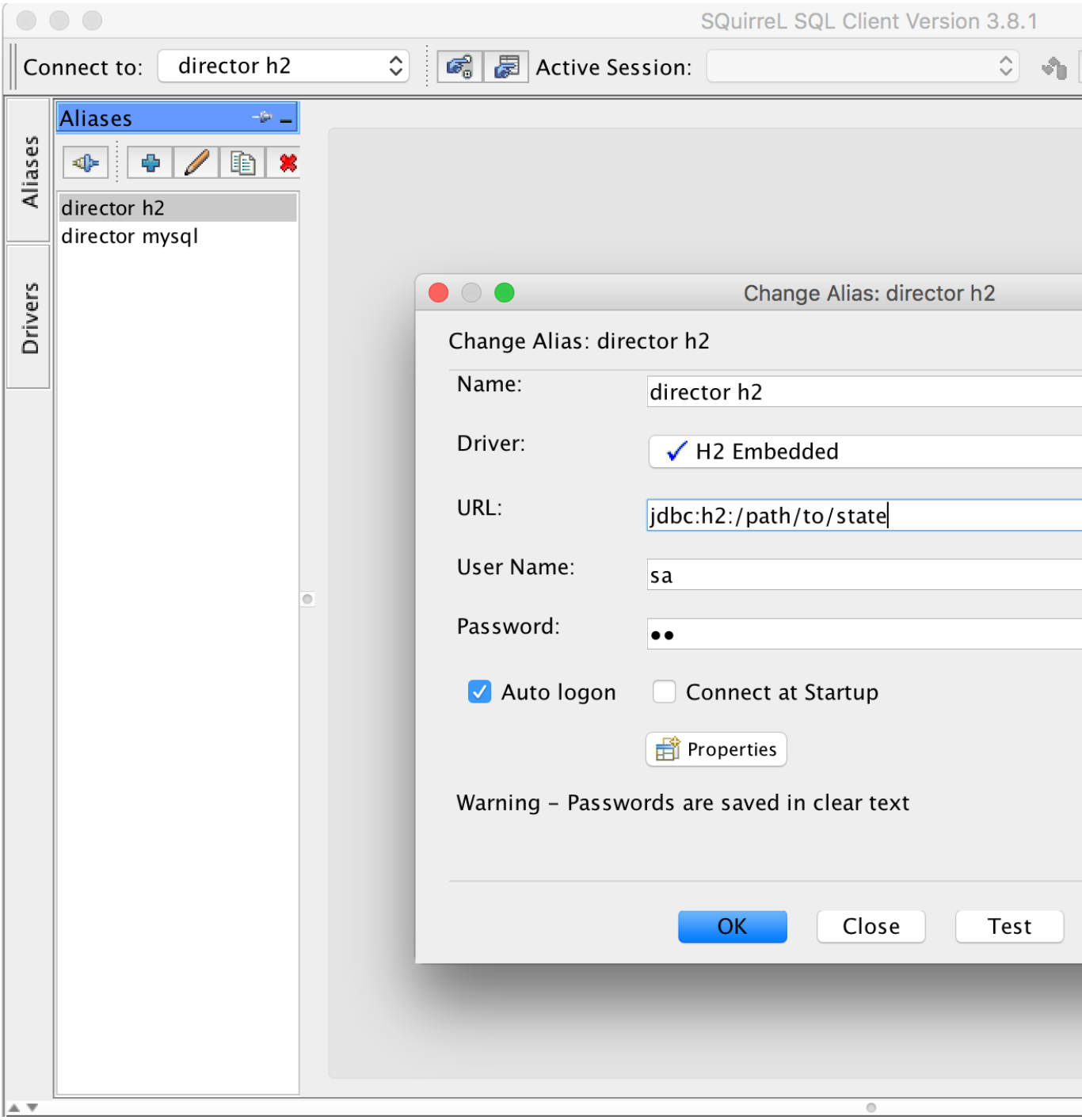
2. Enable the **H2 Embedded** driver in SQuirreL SQL.
   a. Download H2 from http://www.h2database.com/html/download.html.
   b. Edit the **H2 Embedded** driver, adding the downloaded jar file to the **Extra Class Path**.



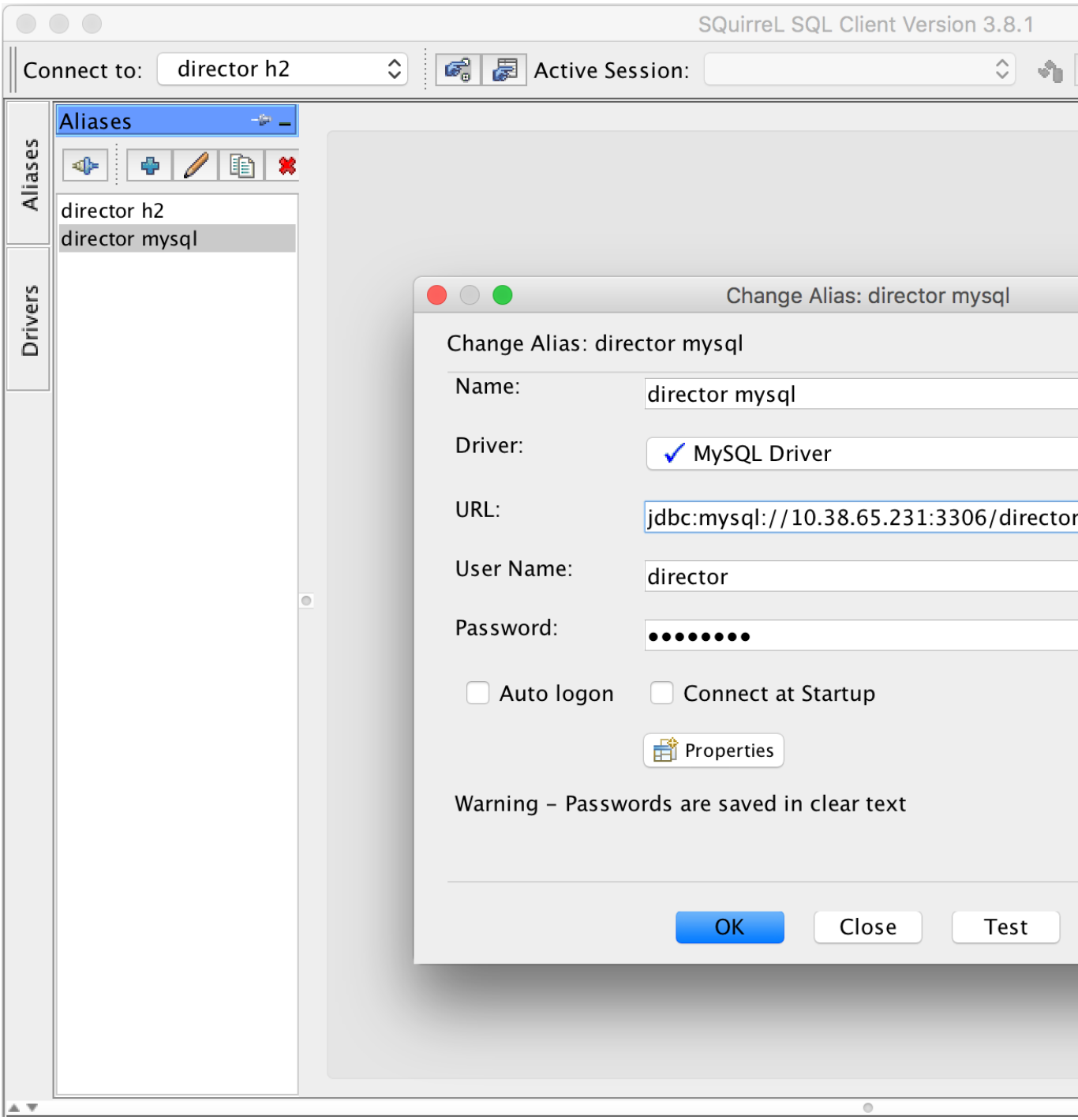
3. Enable the "MySQL Driver" driver in SQuirreL SQL.
   a. Download the MySQL driver from the MySQL Downloads site.
   b. Edit the "MySQL Driver" driver, adding the MySQL connector jar to the Extra Class Path.



4. Create an alias for the H2 database. Test the connection to make sure you can connect to the database.



5. Create alias for MySQL data export database. Test the connection to make sure you can connect to the database.



6. Prepare the H2 database for data export. H2 generates names for indexes that are longer than what is permitted in MySQL, so you must rename the indexes in H2 to ensure that they do not violate the MySQL length limit. For example:

```
ALTER INDEX DEPLOYMENTS_UNIQUE_PER_ENVIRONMENT_BY_DEPLOYMENT_NAME_INDEX_8 RENAME TO
DEPLOYMENTS_UNIQUE_PER_ENVIRONMENT_BY_DEPLOYMENT_NAME
ALTER INDEX UK_EXTERNAL_DATABASE_SERVERS_UNIQUE_PER_ENVIRONMENT_BY_NAME_INDEX_8 RENAME
 TO UK_EXTERNAL_DATABASE_SERVERS_UNIQUE_PER_ENVIRONMENT_BY_NAME
```

Note: Scroll to the right to see the entire lines in the above code sample.

7. Prepare MySQL for data export.

   a. Depending on the Cloudera Director and MySQL versions you are running, you might need the following step. If you see the error message **invalid default value** after performing the next step, you need to do this. Otherwise, this step is optional.

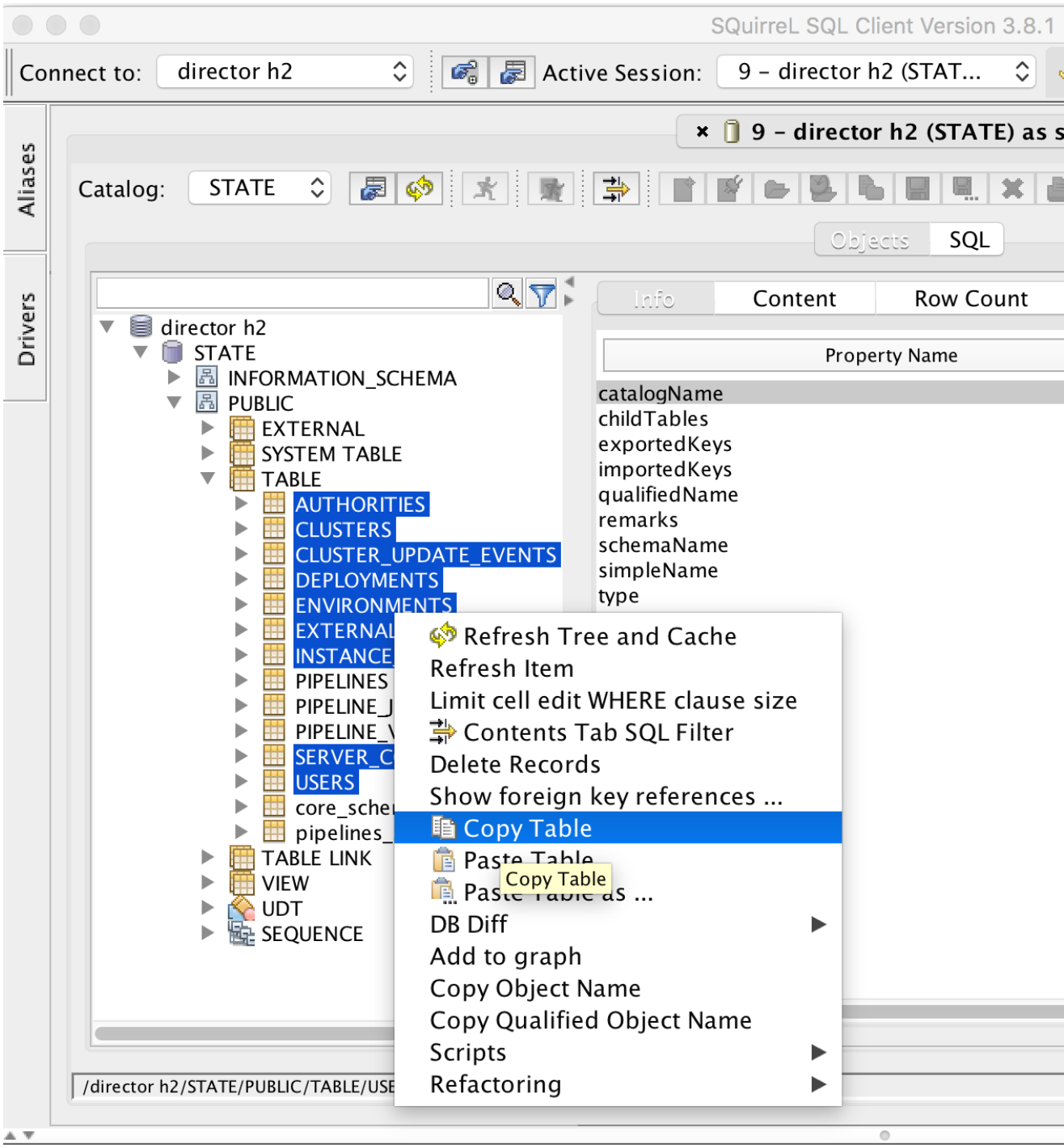
```
select @@global.sql_mode
// Keep the value somewhere if you would like to restore the value
// at the end of this procedure. (See step 9c below.)
set @@global.sql_mode = "...";
// remove NO_ZERO_IN_DATE,NO_ZERO_DATE from previous value.
```

   b. Reconnect to the MySQL alias for this change to apply to your session.

8. Export data from H2 to MySQL.

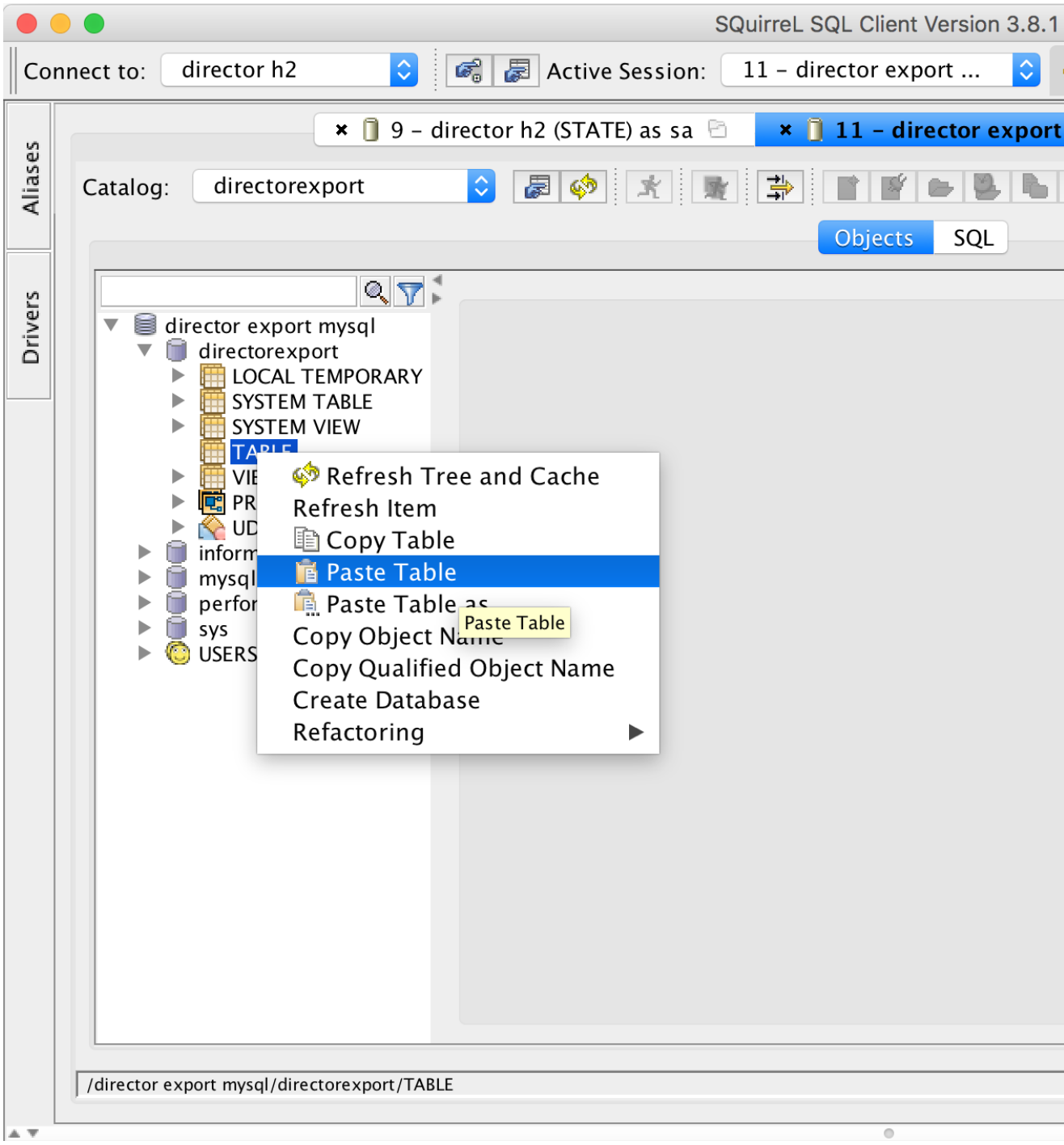
   a. Go to the **Alias** pane in SQuirreL SQL, and connect to H2.

   b. Go to **h2** > **STATE** > **PUBLIC** > **TABLE**. Select all tables except for the `PIPELINES*` and `*schema_versions` tables, right click and select **Copy Table**.



c. Go to the **Alias** pane in SQuirreL SQL and connect to MySQL.

d. Go to **mysql** > **directorexport** > **TABLE**, right click **TABLE** and select **Paste Table**.



9. Restore **sql_mode**, if desired (for more about **sql_mode**, see Prepare MySQL for data export above).

Step 3: Prepare MySQL database for data import

Start Cloudera Director with MySQL.

1. Configure Cloudera Director to use MySQL, as described in Configuring Cloudera Director Server to use the MySQL Database on page 122.
2. Start Cloudera Director with MySQL. When Cloudera Director starts, the database schema will be created.
3. Stop Cloudera Director to prevent modification of the database during data import.

4. Delete all values from the AUTHORITIES, USERS, and SERVER_CONFIGS tables. Cloudera Director populates these tables with some values by default. These values should be deleted so they will not conflict with the imported data.

```
DELETE FROM AUTHORITIES;
DELETE FROM USERS;
DELETE FROM SERVER_CONFIGS;
```

Step 4: Import data to MySQL

1. Dump the data only (no schema) from the export database. You can use the `-h` option if running **mysqldump** against a remote host.

```
$ mysqldump -u [user] -p --no-create-info directorexport > directorexport.sql
```

2. Import the data into Cloudera Director's database. You may use the `-h` option if running **mysql** against a remote host.

```
$ mysql -u [user] -p director < directorexport.sql
```

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 143.

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 131 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_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/mysqld.log
pid-file=/var/run/mysqld/mysqld.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.

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: • `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:

```
ZGVmYXVsdGRpcmVjdG9yZGVzZWRla2V5
```

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

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

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

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

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

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

```
desede;ZGVmYXVsdGRpcmVjdG9yZGVzZWRla2V5|desede;new-key-in-base64
```

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

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

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

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

Wait for the Server to Complete Ongoing Work

Do not try to change encryption while the server is performer 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 Amazon S3 Object Storage

For clusters running on AWS, Amazon S3 (Simple Storage Service) provides an efficient and cost-effective cloud storage option. For information on the uses of Amazon S3 in a CDH cluster, and how to configure Amazon S3 using Cloudera Manager, see How to Configure AWS Credentials and Configuring the Amazon S3 Connector in the Cloudera Enterprise documentation. For links to more topics focused on Amazon S3 from the core Cloudera Enterprise documentation library, see Get Started with Amazon S3.

Note: S3Guard is a feature that guarantees consistent read operations for data stored in Amazon S3. Without S3Guard, Amazon S3 only guarantees "eventual consistency" for data stored in S3, which means that data written to Amazon S3 may not be immediately available for queries and listing operations. S3Guard adds an additional metadata store using an Amazon DynamoDB instance that allows for consistent read operations and improved performance. Instructions for configuring S3Guard in Cloudera Director are included below. For more information on S3Guard, see Configuring and Managing S3Guard in the Cloudera Enterprise documentation.

Important: CDH 5.11.1 clusters using S3Guard can share Amazon DynamoDB tables with CDH 5.12.x clusters, but CDH 5.11.0 clusters *cannot* share tables with CDH 5.12.x clusters. You must upgrade CDH 5.11.0 clusters using S3Guard to CDH 5.11.1 or higher in order to use the same tables for S3Guard metadata as CDH 5.12 clusters.

Configuring Amazon S3 with Cloudera Director

Cluster access to Amazon S3 storage can be configured through Cloudera Director by launching your cluster with a configuration file and the bootstrap-remote CLI command. Cloudera Director will make the necessary API calls and pass your AWS access key information or IAM role information to Cloudera Manager so that S3 access is set up according to your configuration settings. Sample content for the sections of the configuration file needed to configure Amazon S3 access is in the aws.reference.conf configuration file, but is commented-out by default. To provide your cluster instances with access to Amazon S3, configure the following sections of the configuration file:

1. First, create an external account with AWS access in the **External Accounts** section of your configuration file. There are two choices for authentication, as described in the configuration file comments, **AWS access key authentication** or **IAM role authentication**.
   - To use AWS access key authentication, uncomment the appropriate section shown below and provide an AWS access key and an AWS secret key.
   - To use IAM role authentication, uncomment the appropriate section show below and choose or create an IAM policy that includes Amazon S3 access (such as the AWS-managed policy **AmazonS3FullAccess**) and attach this policy to the IAM role that you assign to your cluster instances. IAM roles for instances that will

use S3Guard should also include a policy that gives access to DynamoDB (such as the AWS-managed policy **AmazonDynamoDBFullAccess**). Specify the IAM role for the instance with the `iamProfileName` property in the `common-instance-properties` section of the configuration file.

```
  #
  # External accounts
  #

  # # Any external accounts that should be set up within Cloudera Manager. These will
allow some cluster
  # # services to utilize cloud functionality, such as object stores.
  #
  # # Note: CM/CDH 5.10 is required for this feature. At the moment, only AWS external
 accounts are supported.
  # externalAccounts {
  #
  #     # External account that uses AWS Access Key Authentication. This type of
authentication
  #     # will also require the AWS_S3 service.
  #     AWSAccount1 {
  #         type: AWS_ACCESS_KEY_AUTH
  #         configs {
  #             aws_access_key: REPLACE-ME
  #             aws_secret_key: REPLACE-ME
  #
  #             #
  #           # S3 Guard (added in CM/CDH 5.11) can be enabled to guarantee a consistent
 view of data stored
  #             # in Amazon S3 by storing additional metadata in a table residing in an
 Amazon DynamoDB instances.
  #             # See
https://www.cloudera.com/documentation/enterprise/latest/topics/cm_s3guard.html for more

  #             # details and additional S3 Guard configuration properties.
  #             #
  #
  #             # s3guard_enable: false
  #             # s3guard_region: REPLACE-ME
  #             # s3guard_table_name: s3guard-metadata
  #             # s3guard_table_auto_create: false
  #         }
  #     }
  #
  #     # External account that uses IAM Role Authentication.
  #     AWSAccount2 {
  #         type: AWS_IAM_ROLES_AUTH
  #     }
```

Optionally, to use S3Guard with IAM role authentication, copy the S3Guard configurations from the access key authentication `configs` block to the IAM role authentication section and configure them.

```
# s3guard_enable: false
  # s3guard_region: REPLACE-ME
  # s3guard_table_name: s3guard-metadata
  # s3guard_table_auto_create: false
```

For descriptions of the S3Guard configuration properties, see the table in Configuring S3Guard in the Enterprise documentation. Use the API names given in this table when adding properties to the `configs` block of the Cloudera Director configuration file. For more information about the differences between AWS access key authentication and IAM role-based authentication, and the characteristics and use cases for each of them, see the sections on each in How to Configure AWS Credentials in the Enterprise documentation.

Note: When you bootstrap a cluster with Cloudera Director that is configured to use external accounts, you will see the external accounts in the Cloudera Director web UI. But if you use Cloudera Manager to add external accounts to an existing cluster, while these external accounts will be functional and available to use, they will not appear in the Cloudera Director web UI. In this case, you must use Cloudera Manager to view all external accounts for the cluster.

2. Next, if you are using access key authentication, add (or uncomment) the Cloudera **S3 Connector** service, **AWS_S3**, in the list of cluster services in the **Cluster description** section of the configuration file. You should also add the **AWS_S3** service with IAM role-based authentication if you are enabling S3Guard. Use of IAM role authentication doesn't require adding the AWS_S3 service if S3Guard is not enabled.

```
services: [
                HDFS,
                YARN,
                ZOOKEEPER,
                HBASE,
                HIVE,
                HUE,
                OOZIE,
                SPARK_ON_YARN,
                KAFKA,
                SOLR,
                FLUME,
                IMPALA,
                SQOOP,
                ACCUMULO16,
                KS_INDEXER,
                # SENTRY,    # Sentry requires Kerberos to be enabled
                SPARK2_ON_YARN,
                KUDU,
                # AWS_S3     # Requires Sentry and Kerberos (on default configurations)

              ]
```

3. Finally, point the AWS_S3 service to the external account you created in step #1 above in the **custom service configurations** section :

```
#
   # Optional custom service configurations
   # Configuration keys containing special characters (e.g., '.', ':') must be enclosed
in double quotes.
   #
   # Configuration properties for CDH roles and services are documented at
   #
https://www.cloudera.com/documentation/enterprise/properties/5-11-x/topics/cm_props_cdh5110.html
   #

   #
   # configs {
   #     AWS_S3 {
   #       cloud_account: AWSAccount1
   #     }
   #
   #     HDFS {
   #       dfs_block_size: 134217728
   #     }
   #
   #     MAPREDUCE {
   #       mapred_system_dir: /user/home
   #       mr_user_to_impersonate: mapred1
   #     }
   #
   #     KAFKA {
   #       "num.partitions": 3
   #     }
   # }
```

Note: By default, as noted in the comment in the configuration file shown in step 2 above, Sentry and Kerberos are required for the AWS_S3 service. You can disable this requirement by setting the `key_distribution_policy` to UNSECURE in the `configs` section of the configuration file (shown in step 3 above).

```
AWS_S3 {
  cloud_account: AWSAccount1
  key_distribution_policy: UNSECURE
}
```

Using EBS Volumes for Cloudera Manager and CDH

Cloudera Director 2.2 and higher supports the use of Amazon Elastic Block Store (EBS) volumes with Cloudera Manager and CDH cluster instances. You can use EBS volumes to store HDFS data, stage data for processing, or install other applications. EBS can provide an efficient and cost-effective alternative to S3 or other storage mechanisms.

Note: An advantage of using EBS volumes for cluster storage is that it allows you to pause your cluster and stop the associated EC2 instances during periods of inactivity. You will still be billed for your EBS volumes while the cluster is paused, but will not be billed for the stopped EC2 instances. For information on pausing a cluster, see Pausing a Cluster in AWS on page 57.

Note: If auto-repair is enabled on a cluster that uses Elastic Block Storage (EBS), you must disable auto-repair for the cluster before stopping either the cluster or the Cloudera Manager instance that manages it. For more information about auto-repair, see Auto-Repair for Failed or Terminated Instances on page 104.

EBS Volume Types

Cloudera Director supports the EBS volume types gp2, st1, and sc1:

EBS volume type	Minimum and Maximum Size	Usage
gp2	1 GiB - 16 TiB	General-purpose SSD (solid state drive) volume that balances price and performance for a wide variety of transactional workloads.
st1	500 GiB - 16 TiB	Low-cost HDD (hard disk drive) volume designed for frequently accessed, throughput-intensive workloads.
sc1	500 GiB - 16 TiB	Lowest-cost HDD (hard disk drive) volume designed for less frequently accessed workloads.

For more information, see Amazon EBS Volume Types.

Amazon EC2 Instance Stores

Instance stores, like EBS, provide block storage for EC2 instances, but they cannot be used together with EBS volumes. Instance store volumes are located on disks that are physically attached to the host computer, and they are optionally included with many EC2 instance types.

Important: Cloudera Director does not support using instance store volumes together with EBS volumes for the same EC2 instance. All block storage volumes in an instance should be the same size, capacity, and type.

If an instance type has instance store volumes and you do not specify EBS volumes, Cloudera Director automatically mounts all the instance store volumes that are available. If you *do* specify EBS volumes, Cloudera Director does not mount instance store volumes.

For more information on EC2 instance stores, see Amazon EC2 Instance Stores in the AWS documentation.

Configuring EBS Volumes

You configure EBS volumes in the instance template in the web UI or in the instance section of the configuration file for clusters launched with the CLI and bootstrap-remote. To configure EBS, provide the following information:

- **Number** of EBS volumes you want
- **Type** of the EBS volumes (gp2, st1, or sc1). All EBS volumes for an instance must be of the same type.
- **Size** of the volumes. Specifying a size outside the ranges defined in the table above causes cluster deployment to fail.
- **Encryption**
  - Whether or not to encrypt data in the EBS volume
  - Whether to use the default KMS key for the EBS service or use a custom KMS key

EBS volumes for a Cloudera Manager or CDH cluster instance have the same lifecycle as the instance. EBS volumes are terminated when the instance is terminated. Repair of an instance does not result in the remounting of an existing EBS volume; a new volume is used.

EBS Volume Encryption

Data in EBS volumes can be encrypted at rest. You use two properties for configuring EBS encryption:

- **enableEbsEncryption:** Labeled **Enable EBS Encryption** in the web UI. Set to `true` or `false`. If this value is set to `true`, the data on EBS volumes created with this instance template will be encrypted.
- **ebsKmsKeyId:** Labeled **EBS KMS Key ID** in the web UI. The key used to encrypt data in the EBS volumes. KMS includes a default master key for each service that supports encryption, including EBS. If you leave this field empty, Cloudera Director configures the EBS volumes to use the KMS default master key for EBS. Alternatively, you can import a custom master key from your own key management infrastructure into KMS and specify it here to be used for the EBS service. To specify a custom master key, enter the full Amazon Resource Name (ARN) of the custom master key that you have stored in KMS: `arn:aws:kms:`*`your_key_name`*. For example:

```
arn:aws:kms:us-west-1:635144601417:key/39b8cdf2-923e-721b-9c6c-652a7e517d72
```

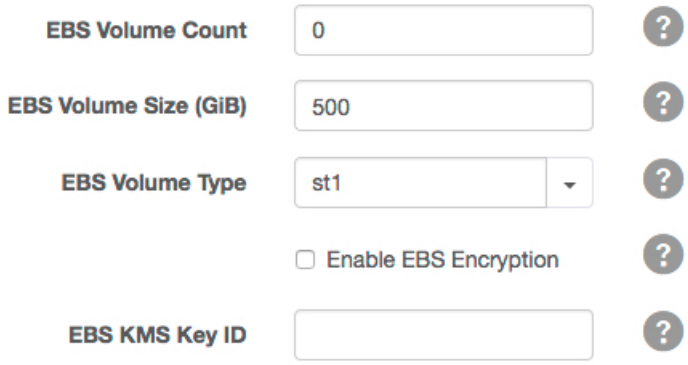
Important: If you specify a custom master key for EBS, you must also add the KMS policy `DescribeKey` to your IAM policy file so that Cloudera Director can validate the custom master key. For more information and a sample IAM policy file that includes `DescribeKey`, see Creating AWS Identity and Access Management (IAM) Policies on page 116.

Note: AWS does not support encryption of root volumes. If you have an EBS root volume and you have enabled EBS encryption, the root volume will not be encrypted.

For more information about EBS encryption, see Amazon EBS Encryption in the AWS documentation. For more information about KMS, see AWS Key Management Service Details in the AWS documentation.

Configuring an EBS Volume with the Web UI

To configure EBS volumes in the web UI, provide the required values in the **Advanced Options** section of the instance template:



Configuring EBS Volumes with the Configuration File

To configure EBS volumes in the configuration file for launching clusters with bootstrap-remote, provide the required values and uncomment them in the **EBS Volumes** section of the file:

```
  #
  # EBS Volumes
  #
  # Director can create and attach additional EBS volumes to the instance. These volumes

  # will be automatically deleted when the associated instance is terminated. These
  # properties don't apply to the root volume.
  #
  # See http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumes.html
  #
  # ebsVolumeCount : 0
  # ebsVolumeType: st1 # specify either st1, sc1 or gp2 volume type
  # ebsVolumeSizeGiB: 500
  #
  # EBS Volume Encryption
  #
  # Encryption can be enabled on the additional EBS volumes. An optional CMK can
  # be specified for volume encryption. Not setting a CMK means the default CMK
  # for EBS will be used. The encryption here does not apply to the root volume.
  #
  # See http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html
  #
  # enableEbsEncryption: false
  # ebsKmsKeyId: arn:aws:kms:REPLACE-ME  # full ARN of the KMS CMK
```

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.

You can also configure Cloudera Manager and CDH services to use Amazon Elastic Block Store (EBS) volumes, as described in Using EBS Volumes for Cloudera Manager and CDH on page 141.

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/v9/environments/{environment}`”, where “`{environment}`” is the name of the environment within which the database server definition is scoped.
- They all use JSON for input data and response data.

Operation	Description	Notes
POST /databaseServers/	Define a new database.	Admin required.
GET /databaseServers/	List all database servers.	
DELETE /databaseServers/{name}	Delete a database server definition.	Admin required.
PUT /databaseServers/{name}	Update a database server definition.	Admin required.
GET /databaseServers/{name}	Get a database server definition.	

Operation	Description	Notes
GET /databaseServers/{name}/status	Get the status of a database server.	
GET /databaseServers/{name}/template	Get the template from which a database server was defined.	

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

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

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

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

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:

- Currently, 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.
- Storage encryption for RDS instances is not supported in Cloudera Director 2.1.x and lower. Storage encryption is supported in Cloudera Director 2.2 and higher, using the default key ID associated with RDS for the AWS account. Use of a nondefault KMS key is not supported.

  To enable storage encryption for a new RDS instance, check the Encrypt DB Instance checkbox in the web UI, or include `storageEncrypted: true` for the instance template in a Cloudera Director configuration file.

Creating a Template to Use Amazon RDS as an External Database

To define an external database server to be created on RDS, you use a template just as you would for any other server. However, you do notspecify the host and port; these are determined as the server is created.

- **name** - A unique name for the server in 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 when 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 the database server instance	`db.m3.medium`

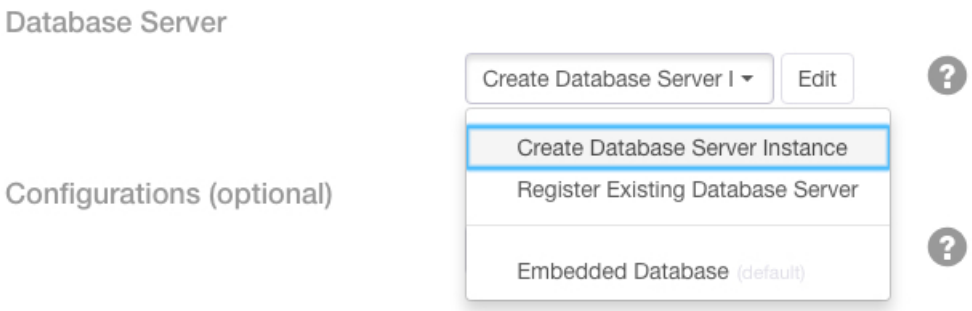
Key	Description	Example
`dbSubnetGroupName`	Name of the database subnet group that the instance spans	`default`
`engineVersion`	(optional) Version of the database engine	5.5.40b
`vpcSecurityGroupIds`	Comma-separated list of security groups for the new instance	`sg-abc123,sg-def456`
`allocatedStorage`	Storage in gigabytes for the new server	`10`
`availabilityZone`	(optional) Preferred availability zone for the new server	`us-east-1d`
`backupRetentionPeriod`	Number of days for which automated backups are retained (0 to disable)	`30`
`skipFinalSnapshot`	Whether to skip a final snapshot before the instance is deleted (`true` to skip; otherwise `false`)	`false`
`storageEncrypted`	Whether stored data on RDS instances is encrypted (`true` to encrypt; otherwise `false`)	`true`

Note:

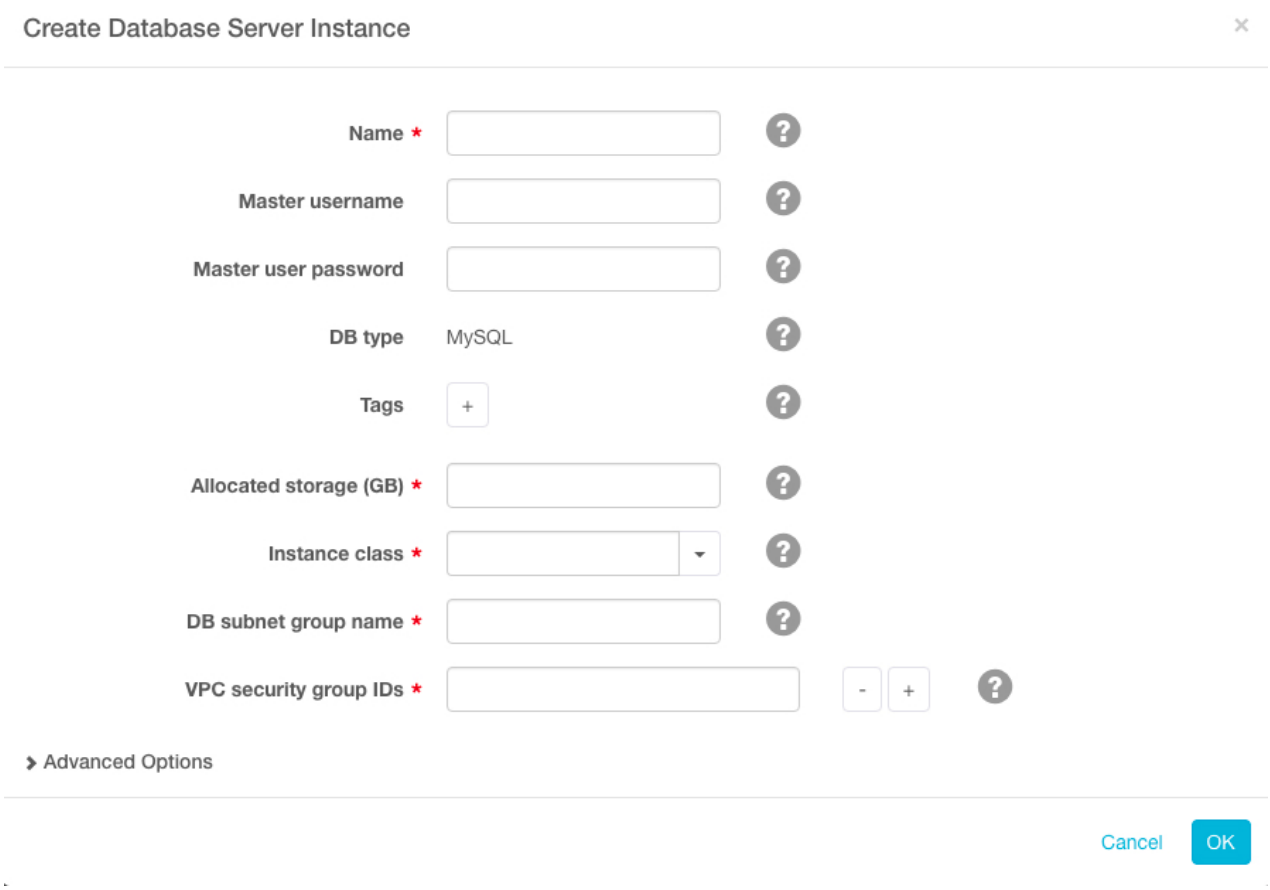
- Cloudera Director does not currently support creating multi-AZ instances.
- The template can also specify tags for the new instance.

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 Cloudera Manager 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**:



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:

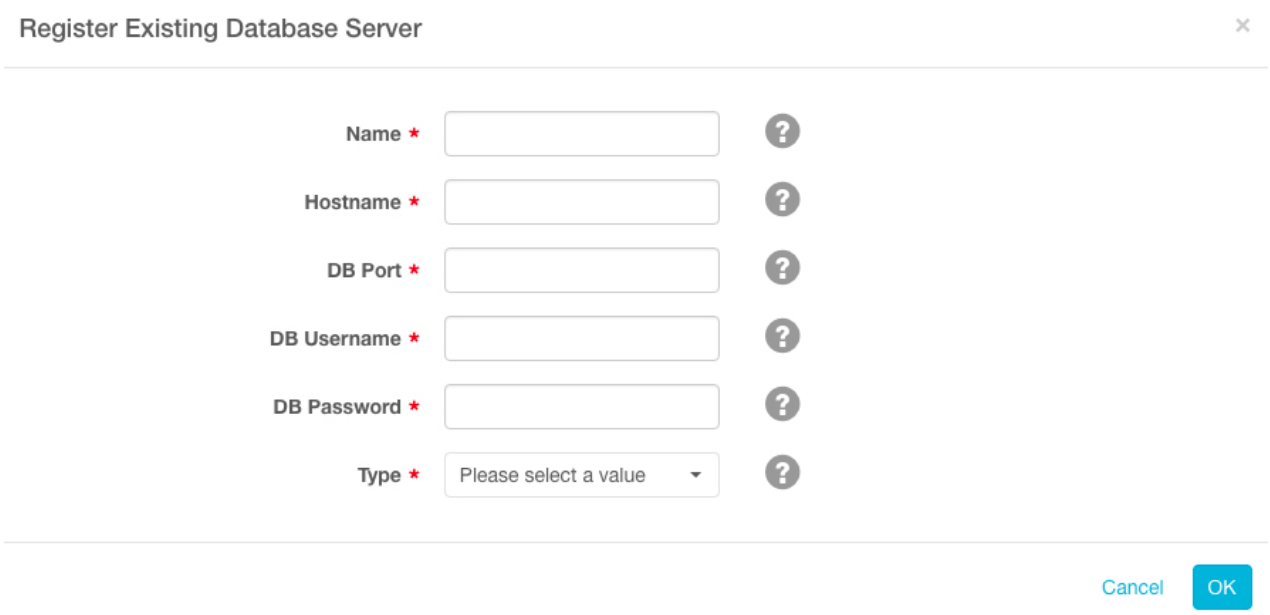


Note: The **DB subnet group name** is not the same as the subnet under the VPC. If the database subnet group name does not exist in the Amazon RDS console, Cloudera Director will fail the validation with the message **DB subnet group not found**.

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 you must create them outside of Cloudera Director and then treat them 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.



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, are eventually 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 can begin bootstrapping new deployments and new clusters that have external database templates that refer to the server. The bootstrap process proceeds in tandem with the server creation, and pauses when necessary to wait for the new RDS instance to be available.

When a deployment or cluster is terminated, Cloudera Director does not terminate the RDS instances. As a result, multiple deployments and clusters can 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 creates 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 creates. 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: navmeta1
         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. Any changed properties in these files take effect automatically when Cloudera Director is restarted. To override existing properties, set new values in the command line.

While the application.properties files are located on the Cloudera Director EC2 instance at `/etc/cloudera-director-client/application.properties` for the client, and `/etc/cloudera-director-server/application.properties`, you can see an example of the files on Cloudera's GitHub site, showing the configuration properties.

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

Pausing Cloudera Director Instances

The Cloudera Director server can be stopped at any time to reduce costs, as can clusters running on Microsoft Azure. These actions are described in the sections below.

Pausing the Cloudera Director Server

Although you can stop and start the Cloudera Director server at any time, you should wait for running workflows to complete.

To start or stop the server, enter the following command:

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

Pausing a Cluster in Azure

There are times when it makes sense to shut down a cluster and stop your Azure virtual machines. The cluster will not be immediately available while stopped and it cannot be used to ingest or process data, but you won't be billed by Microsoft for virtual machines that are stopped and deallocated. Provisioned premium storage disks, and space used by standard storage disks will continue to accrue charges.

Pausing a cluster requires using premium or standard storage disks for all storage, both on management and worker nodes. *Data stored on ephemeral disks will be lost after a cluster pause.*

Important: Any data that is persisted to the ephemeral drive, typically mounted at `/mnt/resource`, will be lost when the virtual machine is shut down. Using HDFS as an example, if `/mnt/resource` is included in the listing for `dfs.datanode.data.dir/dfs_data_dir_list` you will lose block replicas from HDFS, and if you shut down three or more virtual machines, you will lose all replicas of some percentage of blocks. If `dfs_name_dir_list/dfs.namenode.name.dir` includes `/mnt/resource` you will lose the `fsimage` file which is critical to the function of the HDFS NameNode.

Note: The Microsoft Azure Portal, CLI, or API can be used for Azure actions; shutting down the virtual machine locally with the **shutdown** command will leave the virtual machine in a billable state. The Cloudera Manager UI or start and stop API commands can be used for cluster actions.

Shutdown procedure

To pause the cluster, take the following steps:

1. Stop the cluster
2. Stop Cloudera Management Service
3. Stop all cluster virtual machines, including the Cloudera Manager host

Startup procedure

To restart the cluster after a pause, the steps are reversed:

1. Start all cluster virtual machines
2. Start Cloudera Management Service
3. Start the cluster

Since the cluster was completely stopped before stopping the virtual machines, the cluster should be healthy upon restart and ready for use.

Considerations

After starting the virtual machines, Cloudera Manager and its agents will be running but the cluster will be stopped. There will be gaps in Cloudera Manager's time-based metrics and charts.

Azure virtual machines that are configured with static private IP addresses will retain their internal IP address and hostname as long as the associated network interface object is not removed, so no reconfiguration of CDH is required after restart. If your nodes were provisioned via Cloudera Director, you should confirm that the IP configuration of your hosts is set to **static** before shutting down your cluster. The default for the Azure plugin in Cloudera Director 2.4.0 and earlier is **dynamic**. If you need to change the IP configuration from **dynamic** to **static** it can be done via the Azure Portal or PowerShell.

Dynamic public IP addresses will be different after virtual machine shutdown. Static public IP addresses can be configured to remain associated with a stopped virtual machine at additional cost, but it isn't necessary to maintain proper cluster operation.

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 5.12.0 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 Cloudera Director and Cloudera Manager Usage on page 173.

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 labeled `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 by adding the following section to `etc/aws-plugin.conf`:

```
virtualizationMappings {
    customMappingsPath: ec2.customvirtualizationmappings.properties
}
```

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 virtualizaton 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 by adding the following section to `etc/aws-plugin.conf`:

```
ephemeralDeviceMappings {
    customMappingsPath: ec2.customephemeraldevicemappings.properties
}
```

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.

Configuring Cloudera Director to Use Custom Tag Names on AWS

In addition to user-specified tags, Cloudera Director automatically adds certain tags to the AWS resources it creates, including EC2 and RDS instances, EBS volumes, and Spot instance requests. Where applicable, the added tags include

the resource name, the unique identifier that Cloudera Director assigns to the resource, and the name of the template that was used to create the resource.

In some environments, the tag names that Cloudera Director uses may conflict with tag names used for other purposes, so it is possible to customize the tag names. To do so, add the following section to `etc/aws-plugin.conf`:

```
customTagMappings {
   Name: custom-tag-name-for-resource-name,
   Cloudera-Director-Id: custom-tag-name-for-resource-id,
   Cloudera-Director-Template-Name: custom-tag-name-for-resource-template-name
}
```

Using the New Mappings

Once the custom tag mappings have been added to the configuration file, restart the Cloudera Director server so that they are detected and used.

Note: If you have existing resources created by Cloudera Director using the default tag mappings, Cloudera Director will not be able to use their tags correctly once it restarts with the custom tag mappings. You should shut down the Cloudera Director server, update the tags manually using the AWS console or CLI, and then start the Cloudera Director server again.

Post-Creation Scripts

There are three kinds of post-creation scripts, depending on whether they are for the Cloudera Manager deployment, for a CDH cluster as a whole, or for each instance in a CDH cluster. The scripts can be written in any scripting language that can be interpreted on the system where it runs.

Each type of post-creation script can be specified either by embedding scripts in the configuration file or by including paths to script files on the local filesystem:

- **Embedding in the configuration file:** For deployment post-creation scripts, include your script in the `postCreateScripts` section within the cloudera-manager {} configuration block. For cluster post-creation scripts, include your script in the `postCreateScripts` or `instancePostCreateScripts` section within the cluster {} configuration block. These blocks can take an array of scripts, similar to the `bootstrapScript` that can be placed inside the instance {} configuration block.
- **Include paths to files on the local filesystem:** For deployment post-creation scripts, include the path to a script in the `postCreateScriptsPaths` section within the cloudera-manager {} configuration block. For cluster post-creation scripts, include your script in the `postCreateScripts` or `instancePostCreateScripts` section within the cluster {} configuration block. You can provide 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.

Post-creation scripts are available through the configuration file or the Cloudera Director API, but not through the Cloudera Director web UI.

Deployment Post-creation Scripts

Deployment-level post-creation scripts run as root on the Cloudera Manager instance when Cloudera Manager deployment is completed. They are configured in the `cloudera-manager` section of the configuration file in the section `postCreateScripts`.

Deployment-level post-creation scripts can be used to customize the Cloudera Manager instance after a cluster has been created, for example, to add a package or modify a file on the Cloudera Manager instance.

Multiple post-creation scripts can be supplied. They will run in the order they are listed in the configuration file.

The following code block is an excerpt from the reference configuration file on the Cloudera github site:

```
cloudera-manager {
...
    postCreateScripts: ["""#!/bin/sh

# This is an embedded post-creation script that runs as root and can be used to
# customize the Cloudera Manager instance after the deployment has been created.

# If the exit code is not zero Cloudera Director will fail

# Post-creation scripts also have access to the following environment variables:

#    DEPLOYMENT_HOST_PORT
#    ENVIRONMENT_NAME
#    DEPLOYMENT_NAME
#    CM_USERNAME
#    CM_PASSWORD

echo 'Hello World!'
exit 0
    """,
    """#!/usr/bin/python

# Additionally, multiple post-creation scripts can be supplied.  They will run
# in the order they are listed here.  Interpeters other than bash can be used
# as well.

print 'Hello again!'
    """]

    # For more complex scripts, post-creation scripts can be supplied via local
    # filesystem paths. They will run after any scripts supplied in the previous
    # postCreateScripts section.
    # postCreateScriptsPaths: ["/tmp/test-script.sh",
    #                          "/tmp/test-script.py"]
...
}
```

Cluster Post-creation Scripts

There are two types of cluster post-creation scripts:

- A *cluster-level script* is run on a single arbitrary instance in the cluster.
- An *instance-level script* is run on each instance in the cluster.

As with deployment-level scripts, cluster post-creation scripts can be specified either by embedding scripts in the configuration file or by including paths to script files on the local filesystem. For both instance-level and cluster-level scripts (and *unlike* `bootstrapScript` and `bootstrapScriptPath`), both post-creation scripting methods can be used simultaneously. For example, `postCreateScripts` could be used for setup (package installation, light system configuration), and `postCreateScriptsPaths` could be used to refer to more complex scripts that may depend on the configuration that was performed in `postCreateScripts`.

Instance-level and cluster-level scripts run when bootstrapping is complete and the cluster is ready. Instance-level scripts will also be run when you grow a cluster by adding instances, but will not run when instances are migrated manually. For instance-level and cluster-level scripts, where there can be multiple instancePostCreateScripts and postCreateScripts, the scripts run in the following order:

1. Everything in the `instancePostCreateScripts` block is run sequentially.
2. Everything in `instancePostCreateScriptsPaths` is run sequentially.
3. Everything in the `postCreateScripts` block is run sequentially.
4. Everything in `postCreateScriptsPaths` is run sequentially.

Cluster-level Post-creation Scripts

Cluster-level post-creation scripts run as root on a single arbitrary instance in a cluster after the cluster has been created. As with instance-level post-creation scripts, they are configured in the `cluster` section of the configuration file. They run after any instance post-creation scripts.

The following code block is an excerpt from the reference configuration file on the Cloudera github site:

```
cluster {
...
    postCreateScripts: ["""#!/bin/sh

# This is an embedded post-creation script that runs as root and can be used to
# customize the cluster after it has been created. This will run only once,
# at a cluster level, on an arbitrary cluster instance.

# If the exit code is not zero Cloudera Director will fail

# Post-creation scripts also have access to the following environment variables:

#    DEPLOYMENT_HOST_PORT
#    ENVIRONMENT_NAME
#    DEPLOYMENT_NAME
#    CLUSTER_NAME
#    CM_USERNAME
#    CM_PASSWORD

echo 'Hello World!'
exit 0
    """,
    """#!/usr/bin/python

# Additionally, multiple post-creation scripts can be supplied.  They will run
# in the order they are listed here.  Interpeters other than bash can be used
# as well.

print 'Hello again!'
    """]

    # For more complex scripts, post-creation scripts can be supplied via local
    # filesystem paths. They will run after any scripts supplied in the previous
    # postCreateScripts section.
    # postCreateScriptsPaths: ["/tmp/test-script.sh",
    #                          "/tmp/test-script.py"]
}
```

Instance-level Post-creation Scripts

Instance-level post-creation scripts run as root after a cluster has been created. They are configured in the `cluster` section of the configuration file.

They run before any cluster-level post-creation scripts. Instance-level post-creation scripts can be used, for example, to specify processes that have to be run separately on each instance, such as to add a package to all cluster instances or modify a file on all cluster instances.

The following code block is an excerpt from the reference configuration file on the Cloudera github site:

```
cluster {
...
    instancePostCreateScripts: ["""#!/bin/sh

# This is an embedded instance post-creation script that runs as root and can be used
to
# customize each cluster instance after the cluster has been created. This script will
 run
# on every cluster instance. These scripts run before postCreateScripts, which are at
cluster level.

# If the exit code is not zero Cloudera Director will fail
```

```
# Instance post-creation scripts also have access to the following environment variables:

#    DEPLOYMENT_HOST_PORT
#    ENVIRONMENT_NAME
#    DEPLOYMENT_NAME
#    CLUSTER_NAME
#    CM_USERNAME
#    CM_PASSWORD

echo 'Hello World!'
exit 0
    """,
    """#!/usr/bin/python

# Additionally, multiple instance post-creation scripts can be supplied.  They will run
# in the order they are listed here.  Interpeters other than bash can be used
# as well.

print 'Hello again!'
    """]

   # For more complex scripts, instance post-creation scripts can be supplied via local

    # filesystem paths. They will run after any scripts supplied in the previous
    # instancePostCreateScripts section.
    # instancePostCreateScriptsPaths: ["/tmp/test-script.sh",
    #                          "/tmp/test-script.py"]
...
}
```

Predefined Environment Variables

As noted in the code comments above, 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.

Deployment-level post-creation scripts do not use the cluster name variable, since they can include multiple clusters.

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	director_environment	The name of the environment that this cluster belongs to.
DEPLOYMENT_NAME	director_deployment	The name of the Cloudera Manager deployment that this cluster belongs to.
CLUSTER_NAME	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. **Note:** This variable is not available for deployment-level post-creation scripts, since Cloudera Manager deployments are not necessarily associated with a particular cluster.

Variable Name	Example	Description
CM_USERNAME	admin	The username needed to connect to the Cloudera Manager deployment.
CM_PASSWORD	admin	The password needed to connect to the Cloudera Manager deployment.

Enabling TLS with Cloudera Director

Transport Layer Security (TLS) is a security protocol that supersedes Secure Sockets Layer (SSL). It is designed to prevent eavesdropping, tampering, and message forgery by encrypting network communications. It also supports authentication of host certificates prior to encryption, to prevent spoofing. You can enable TLS on your clusters, as well as on Cloudera Manager and Cloudera Director, in order to protect communications among them.

Cloudera Director supports TLS, but with the following limitations:

- Cloudera Director can be configured to require TLS for access, so that communications between its server and clients are secured. However, the generated Java and Python client libraries for the Cloudera Director server API, which are provided in the Cloudera Director SDK, cannot communicate with a Cloudera Director server that is running under TLS. Also, the Cloudera Director CLI cannot communicate with a Cloudera Director server that is running under TLS for operations such as `bootstrap-remote`.
- Cloudera Director cannot directly enable TLS in Cloudera Manager deployments that it bootstraps. Instead, see Configuring Cloudera Manager Clusters for TLS/SSL in the Cloudera Manager documentation for instructions on enabling TLS.

  Important: Once Cloudera Manager is configured with TLS, even at **Level 0**, Cloudera Director will no longer be able to communicate with it.

- Cloudera Director can be used to configure TLS for a cluster's services by implementing the necessary steps, using features such as bootstrap scripts and/or designing instance images that have required files, such as truststores, already in place. See Configuring TLS/SSL Encryption for CDH Services in the Cloudera Security documentation for instructions on configuring TLS for CDH services. While Cloudera Manager can be used independently of Cloudera Director to set up TLS for a cluster, Cloudera Director would then not know to perform the same procedures on new instances created during grow or clone operations.

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.

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.

Note: To deploy a cluster that connects to a special Kerberos appliance such as FreeIPA, run the following command on the Cloudera Manager host to skip the credential import:

```
sed -i '/kinit/i exit 0' /usr/share/cmf/bin/import_credentials.sh
```

This step is required because Cloudera Manager cannot import the credentials, and the credentials are not necessary in this scenario.

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, at the time of cluster setup.

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.

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 *principal@your.KDC.realm*
krbAdminPassword	The password for the administrative Kerberos account.
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	Active Directory suffix where all the accounts used by CDH daemons will be created. Used only if Active Directory KDC is being used for authentication. This configuration should be in the format of an X.500 Directory Specification (DC=domain,DC=example,DC=com).
KRB_MANAGE_KRB5_CONF	Set this to `true`. This allows Cloudera Manager to deploy Kerberos configurations to cluster instances. The value `false` 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 hdfsmasters-1, hdfsmasters-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 hdfsmasters-1 instances, one hdfsmasters-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.

  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.
  - One HUE_LOAD_BALANCER role

  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.

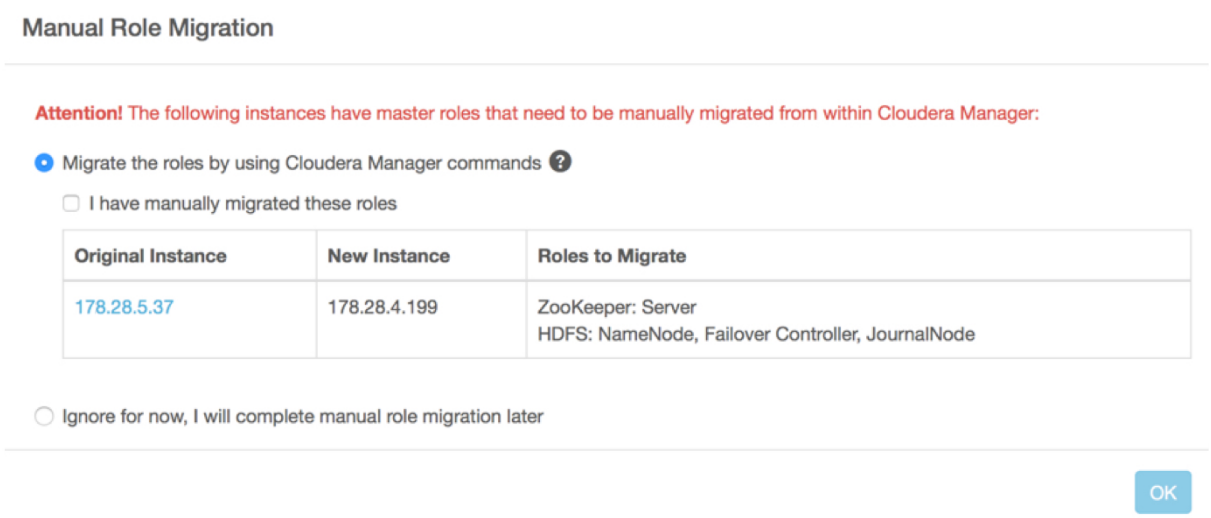
  For more information, see Oozie High Availability in the Cloudera Administration documentation.
- The following requirements apply to databases for your cluster:
  - You can configure external databases for use by the services in your cluster and for Cloudera Director. If no databases are specified in 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.
- Instance-level post-creation scripts will not be run on any instances that are part of a manual migration. If you have instance-level post-creation scripts and want them to run during manual migration, run the scripts manually.
- 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:



- 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**.
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.

Cloudera Director and Cloudera Manager Usage

Cloudera Director works with Cloudera Manager and the cloud service provider to provide centralized and programmatic administration of clusters in the cloud, including deployment, configuration, and maintenance of CDH clusters. With Cloudera Director, you can monitor and manage multiple Cloudera Manager and CDH deployments, across different cloud environments.

When you use Cloudera Director to deploy CDH, you can perform administrative tasks either in Cloudera Director or in Cloudera Manager. To avoid conflicts and inconsistencies, use the management tool most appropriate for the task.

- With Cloudera Director 2.4 and higher and Cloudera Manager and CDH 5.11 and higher, many changes made directly in Cloudera Manager are detected by Cloudera Director, which periodically refreshes its state to reflect the state of the cluster in Cloudera Manager. Cloudera Director also refreshes its stored templates for the cluster so that your updated configuration is used if you create more instances or clone the cluster.

  Note: After making modifications in Cloudera Manager, wait at least five minutes for Cloudera Director to refresh before making any cluster modifications in Cloudera Director-side cluster modifications, such as grow or shrink.

- With Cloudera Director 2.3 and lower or Cloudera Manager and CDH 5.10 or lower, if you perform certain administrative tasks in Cloudera Manager, Cloudera Director and Cloudera Manager will become out of sync. When Cloudera Director and Cloudera Manager are out of sync, Cloudera Director cannot grow or shrink the cluster or perform other updates to the cluster. You can use Cloudera Director 2.3 and lower to deploy new Cloudera Manager instances and clusters, but Cloudera Manager instances that are out of sync with Cloudera Director will function independently of Cloudera Director.

When to Use Cloudera Director

Use Cloudera Director when you want to perform the following types of tasks:

- Deploying Cloudera Manager and CDH clusters for prototyping.
- Deploying Cloudera Manager and CDH clusters when you have finalized the topology and configuration.
- Growing or shrinking a cluster. If you have made changes to the cluster using Cloudera Manager, update Cloudera Director with the changes and redeploy the cluster before you grow or shrink the cluster.
- Setting up clusters with Kerberos authentication or high availability.

When to Use Cloudera Manager

Use Cloudera Manager when you want to perform the following types of tasks:

- Adding or removing a service in an existing cluster.
- Changing role assignments for an existing virtual instance group, or migrate roles from one instance to another
- Changing the configuration of a service or role
- Testing and iterating on the topology and configuration of clusters.

  Use Cloudera Director to create the cluster when you have finalized the topology and configuration.

- Setting up TLS and wire encryption.

When encryption is enabled, Cloudera Director cannot communicate with Cloudera Manager.

CDH Cluster Management Tasks

When you deploy CDH and Cloudera Manager through Cloudera Director, you use Cloudera Director or Cloudera Manager to manage the clusters, depending on the task.

The following table lists cloud administrative tasks and the application where you must perform them to avoid inconsistencies in Cloudera Director and Cloudera Manager:

Task	Application	Notes
Cluster setup	Cloudera Director	Cloudera Director cannot manage clusters that are set up directly in Cloudera Manager.
Addition of host to a cluster or addition of cluster to Cloudera Manager	Cloudera Director	
Host decommission	Cloudera Director	This is done by deleting the instance from the virtual instance group using Cloudera Director.
Adding a service	Cloudera Manager	If you add a service to a cluster in Cloudera Manager, Cloudera Director will detect the change and will update its cluster template to match.
Removing a service	Cloudera Manager	If you remove a service from a cluster in Cloudera Manager, Cloudera Director will detect the change and will update its cluster template to match. You can stop a service instead of removing it from a cluster. You can also use the grow and shrink feature of Cloudera Director to create hosts that do not have that service's roles.
Initial role assignment	Cloudera Director	
Add new virtual instance groups to a cluster	Cloudera Director	
Change role assignments for an existing virtual instance group, or migrate roles from one instance to another	Cloudera Manager	Cloudera Director periodically refreshes its data on the state of cluster roles in existing virtual instance groups to include changes made with Cloudera Manager.
Changes to the configuration of a service or role	Cloudera Manager	Cloudera Director will detect service and role configuration changes made in Cloudera Manager and will update the cluster template and instance templates to match. The changes must not result in inconsistency with respect to the roles included in different instances of the same virtual instance group. See Ensuring Consistency of Virtual Instance Groups on page 176 below for more information.
Cloud provider settings for instances, such as the machine image or instance type	Cloudera Director	Cloudera Director will detect configuration changes made in the cloud provider, including changes in the instance type and machine image, and will update the cluster template and instance templates to match. The changes must not result in inconsistency with respect to the cloud provider configuration included in different instances of the same virtual instance group. See Ensuring Consistency of Virtual

Task	Application	Notes
		Instance Groups on page 176 below for more information.
Adding, removing, and modifying parcels	Cloudera Manager	If you activate or deactivate parcels in Cloudera Manager, Cloudera Director will detect this change and update its cluster template to match. Parcel version changes will also be detected. Note that when deactivating a parcel in Cloudera Manager, the associated services for that parcel should also be removed through Cloudera Manager.
Cloudera Manager username and password change	Cloudera Manager and Cloudera Director	Change the username and password in Cloudera Manager. After you change the username and password in Cloudera Manager, you must update the information in Cloudera Director. If you do not update the information in Cloudera Director, Cloudera Director will not be able to monitor or modify the cluster.
Upgrading a Cloudera Manager license	Cloudera Manager	Use Cloudera Manager to upgrade from Cloudera Express to Cloudera Enterprise. Cloudera Director will not display the state of the updated license, but will not prevent Cloudera Enterprise functionality.
Minor version upgrade to Cloudera Manager or CDH	Cloudera Manager	You must upgrade Cloudera Manager manually and then use the upgraded Cloudera Manager to upgrade CDH. Cloudera Director will detect the version changes, and new clusters will use the upgraded versions.
Enabling high availability during cluster setup	Cloudera Director	High availability is supported in Cloudera Director version 2.0 or higher. Use the configuration file to enable high availability. Do not use the Cloudera Director web UI.

Task	Application	Notes
Enabling high availability in an existing cluster	Cloudera Manager	See High Availability in the Cloudera Manager documentation for more information. **Note:** When Cloudera Director 2.4 and higher refreshes a cluster to incorporate changes made in Cloudera Manager, it detects changes made at the role group level, but it does not detect configuration changes made at the role instance level. In some cases with Cloudera Manager 5.11 and earlier, the Cloudera Manager high availability (HA) wizard introduces role instance level configuration changes, and these must be moved manually to role group configurations in Cloudera Manager to keep Cloudera Director in sync. This is not an issue if you are using Cloudera Manager 5.12 and higher.
Modifying a cluster in a highly available deployment	Cloudera Director	If you enable high availability in Cloudera Manager, you can run modify operations only on instance groups that do not contain highly available master roles.
Enabling Kerberos authentication during cluster setup	Cloudera Director	Kerberos setup is supported in Cloudera Director version 2.0 or higher. Use the configuration file to enable Kerberos. Do not use the Cloudera Director web UI. If you use Cloudera Director to deploy a cluster but use Cloudera Manager to enable Kerberos authentication, Cloudera Director and Cloudera Manager will become out of sync.

Ensuring Consistency of Virtual Instance Groups

All instances in a virtual instance group must have identical roles assigned to them, with identical role configurations, in order to enable cluster modifications in Cloudera Director. When using Cloudera Director 2.4 and above with Cloudera Manager and CDH 5.11 and above, you can change role assignments and role configurations in Cloudera Manager, but you must ensure that all instances in a given virtual instance group are configured identically. Cloudera Director will then propagate any changes you make in Cloudera Manager back into the cluster's instance templates.

If you make changes to an instance that create role assignments or configurations different from those of other instances in the virtual instance group, Cloudera Director will detect the inconsistency and will flag the virtual instance group in the Cloudera Director UI, identifying which instance is inconsistent what the inconsistencies are. You will not be able to grow that instance group until the inconsistency is fixed.

There are two ways to fix inconsistencies in a virtual instance group:

- In Cloudera Manager, assign or remove roles in the instances that are flagged as inconsistent so that they are identical to the other instances in the virtual instance group
- In Cloudera Director, shrink the virtual instance group to remove the instances that are flagged as inconsistent

CDH Cluster Management Guidelines for Cloudera Director

When you use Cloudera Director to deploy Cloudera Manager and CDH, the cluster information is saved in the Cloudera Director database. If you make changes to the cluster using the cloud provider management console, the changes cannot be detected by Cloudera Director. As a result, Cloudera Director will have incorrect information about the configuration and state of the cluster.

Use the following guidelines when you manage CDH clusters deployed through Cloudera Director:

- You cannot update the AMI of an instance in Cloudera Director. When an AMI is scheduled for retirement, you must migrate the nodes in the instance that uses the AMI to a new instance with a new AMI before the AMI is retired. The clusters that use the AMI will not be affected by the AMI retirement. However, growing an instance group that uses the retired AMI will fail because the AMI is no longer available. Use Cloudera Director to migrate the cluster nodes to a new instance with a new AMI.
- Terminating an instance using the cloud provider management console results in poor health of the hosts and services in Cloudera Director. If the health of an instance turns bad or the instance fails, you can migrate to a new instance. Use the Cloudera Director web UI to shrink and grow the worker nodes and migrate the master node to a new instance.
- Cloudera Director does not support resizing or changing the instance type of a deployed instance using the cloud provider management console. Instead, use Cloudera Director's grow and shrink functionality to migrate to a new instance with the appropriate instance type.

For information about growing or shrinking a cluster, see Modifying the Number of Instances in an Existing Cluster on page 183.

For information about migrating HDFS master roles to a new instance, see Using Role Migration to Repair HDFS Master Role Instances on page 169.

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 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 193.
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.

Ports Used by Cloudera Director

Cloudera Director needs to communicate with each of the nodes in the clusters that it manages. The simplest way to achieve this, if your organization's security policies allow it, is to enable all network traffic between Cloudera Director, cluster instances, and the Cloudera Manager node using any protocol on any port. You can do this in AWS by creating a security group for your VPC that allows traffic between its members and assigning this security group to Cloudera Director, Cloudera Manager, and all cluster instances. With this approach, you do not have to specify each port that is required by Cloudera Manager.

Type	Protocol	Port Range	Source
`ALL Traffic`	`ALL`	`ALL`	*security_group_id*
`SSH (22)`	`TCP (6)`	22	0.0.0.0/0

In a restricted network environment, you may want to enable minimal network traffic between instances and keep open ports to a minimum.

- Minimally, open port 22 for traffic to allow SSH access to the Cloudera Director server. If using SSH tunneling, the other Cloudera Director ports below are not required.
- Minimally, the Cloudera Director server needs SSH (port 22) access to every node in the cluster.
- Open outbound port 123 so that the Cloudera Manager and cluster nodes can access an NTP time server.
- Optionally, open port 7189 on the Cloudera Director server to enable access to the Cloudera Director web UI. Optionally, you can configure Cloudera Director to use HTTPS. You can configure a non-default port for the Cloudera Director web UI by adding the `server.port` property to the server `application.properties` file and specifying the desired port number. To enable HTTPS, configure the `server.ssl.*` settings in the SSL section of the `application.properties` file.
- Optionally, open port 7180 on the Cloudera Manager instances so that the Cloudera Director server can use port 7180 to interact with the Cloudera Manager API. (Otherwise, Cloudera Director will use SSH tunnels on port 22 to communicate with Cloudera Manager.)
- The Cloudera Director server needs access to outbound ports 80 and 443 to retrieve packages for initial installation, metering access, and for API access to the AWS, Azure, and Google APIs. Refer to AWS, Azure, and Google documentation for the exact domains.

For information on ports used by Cloudera Manager and CDH, see Ports in the Cloudera Manager documentation.

The following table summarizes the Cloudera Director port requirements described above:

Service	Role	Purpose	Default Port	Protocol	Required?
Cloudera Director	Cloudera Director server	Cloudera Director web UI and API	7189 (configurable)	HTTP	No (SSH tunnel can be used instead)
		Web UI and API	configurable	HTTPS	No (SSH tunnel can be used instead)

Service	Role	Purpose	Default Port	Protocol	Required?
Clusters managed by Cloudera Director	Cloudera Manager node	Cloudera Manager API	7180	HTTP	No (SSH tunnel can be used instead)
		NTP	123 (outbound)	UDP	Yes
		Node installation	22	SSH	Yes
	Cluster nodes	NTP	123 (outbound)	UDP	Yes
		Node installation	22	SSH	Yes
archive.cloudera.com, metering.cloudera.com, AWS, Azure, and Google REST APIs, etc.	Cloudera Director server and the Cloudera Manager node	Software download/metering	80 (outbound)	HTTP	Yes*
			443 (outbound)	HTTPS	Yes*

*You can restrict access to archive.cloudera.com and metering.cloudera.com if you have an internal parcel repository and Cloudera Manager repository, and are not using usage-based billing (which requires metering), but your instances still require access to your cloud provider's REST APIs through HTTP or HTTPS.

SSH Keys in Cloudera Director

The Role of Keys in SSH

SSH is a protocol for communicating over an encrypted channel between computers. A common use of SSH is to connect to a remote computer in order to establish a shell process, within which commands may be issued. The SSH **client** logs in to the SSH **server** and receives a shell prompt. The client can issue shell commands, and then terminate the connection or log out when finished.

SSH uses the same kind of strong encryption that is used by TLS and SSL for encrypted web traffic. An observer should not be able to determine the contents of the network traffic between the client and the server. Through encryption, SSH provides confidentiality of communications.

When establishing an SSH connection for shell use, a client needs to authenticate to a server. Authentication can be performed in a few ways. One common way is through a username and password pair. While this is a well understood and widely supported means of authentication, it is vulnerable to the use of weak passwords.

A better option is to use a key pair. Like key pairs in TLS and SSL, an SSH key pair is composed of a private key and a corresponding public key. When the server is configured with the public key, the client can authenticate using the private key. Due to the way that the authentication mechanism works, the private key does not need to be passed over the network. Also, a private key is much larger, and much harder to guess, than a password. For these reasons, SSH key authentication is almost always preferred over password authentication.

Use of SSH Keys in Cloud Providers

When a cloud provider creates a new instance, and you want to be able to connect to it over SSH, you must somehow inform the provider about the public key for authentication. The cloud provider is responsible for installing the public key in the correct location for it to work for authenticating with the corresponding private key.

The way that you tell a cloud provider about a key pair varies. For example, in AWS, you set up an EC2 key pair, which has an associated name. You can either have AWS generate the key pair and send you the data, or generate the key pair yourself and only upload the public key. No matter the provider, when you create an instance, you include information about the key pair.

Generally, the cloud provider copies the public key into the **authorized_keys** file in the standard login account for the instance. This is enough to configure the instance's SSH server to allow authentication to that account with the corresponding private key. To use it on the command line:

```
ssh -i /path/to/private_key.pem username@instance_ip_address
```

Cloudera Director's Use of SSH Keys

Cloudera Director uses SSH to issue commands to instances that it launches and configures as part of bootstrapping deployments and clusters. Therefore, Cloudera Director requires the private key used to authenticate to the standard login account of each instance that it launches. It is insufficient for it to only be supplied the public key or an identifier for the private key material; only the private key itself is sufficient for authentication to succeed.

When you create a new environment in Cloudera Director, you supply the default username for a standard login account and the default SSH private key for authentication to that account. Every instance that Cloudera Director launches under the environment is accessed, by default, using the username and private key. An instance template, which specifies instance features like size and backing image, may override the username.

The private key must be provided to Cloudera Director in unencrypted form. However, Cloudera Director encrypts its own database where it stores private keys.

Good Practices for SSH Key Management with Cloudera Director

- **Use large key sizes.** When creating an RSA key pair, use a key length of at least 2048 bits.
- **Protect private keys by keeping them off the network.** A cloud provider only requires the public key from an SSH key pair to launch instances you can authenticate to. The corresponding private key does not need to leave the computer where it was generated. Accordingly, for better security, instead of having AWS generate your key pair, generate it locally and upload only the public key.
- **Change the Cloudera Director database encryption password.** Cloudera Director ships with encryption of its own database enabled, but with a default encryption password. Follow the instructions for changing the encryption password in Starting with Encryption to avoid easy decryption if the database is compromised.
- **Do not use personal SSH key pairs for Cloudera Manager and cluster instances.** Instead, generate dedicated "service" SSH key pairs that are only for use by instances launched by Cloudera Director. This allows the lifecycle of personal and service key pairs to be managed separately. Also, service key pairs may then be shared without compromising any individual user's key pairs.
- **Install additional SSH public keys on instances with bootstrap scripts.** To permit wider authentication to instances, use a bootstrap script to append approved public keys to the **authorized_keys** file for the standard login accounts, as an alternative to sharing the private key of the "primary" SSH key pair installed by the cloud provider.

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.numberOfHealthCacheExecutorThreads`** - The number of threads used to simultaneously request health information from Cloudera Manager. the default value is 5.
- **`lp.cache.health.expirationMultiplier`** - Used to determine if a health value is stale. If the health value has not been updated in `pollingRateInMilliseconds * expirationMultiplier` milliseconds, then the health value is considered stale and is reported to the 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:

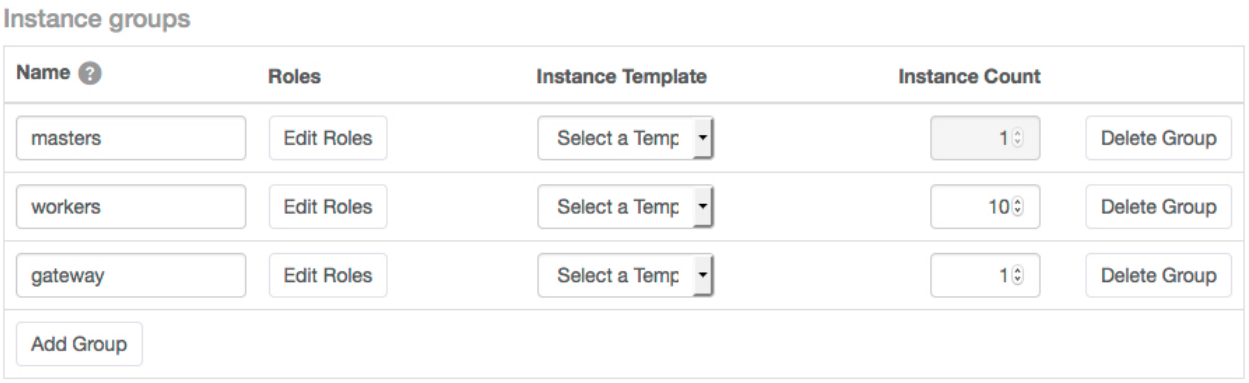
- For HDFS - DataNode
- For YARN - NodeManager

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

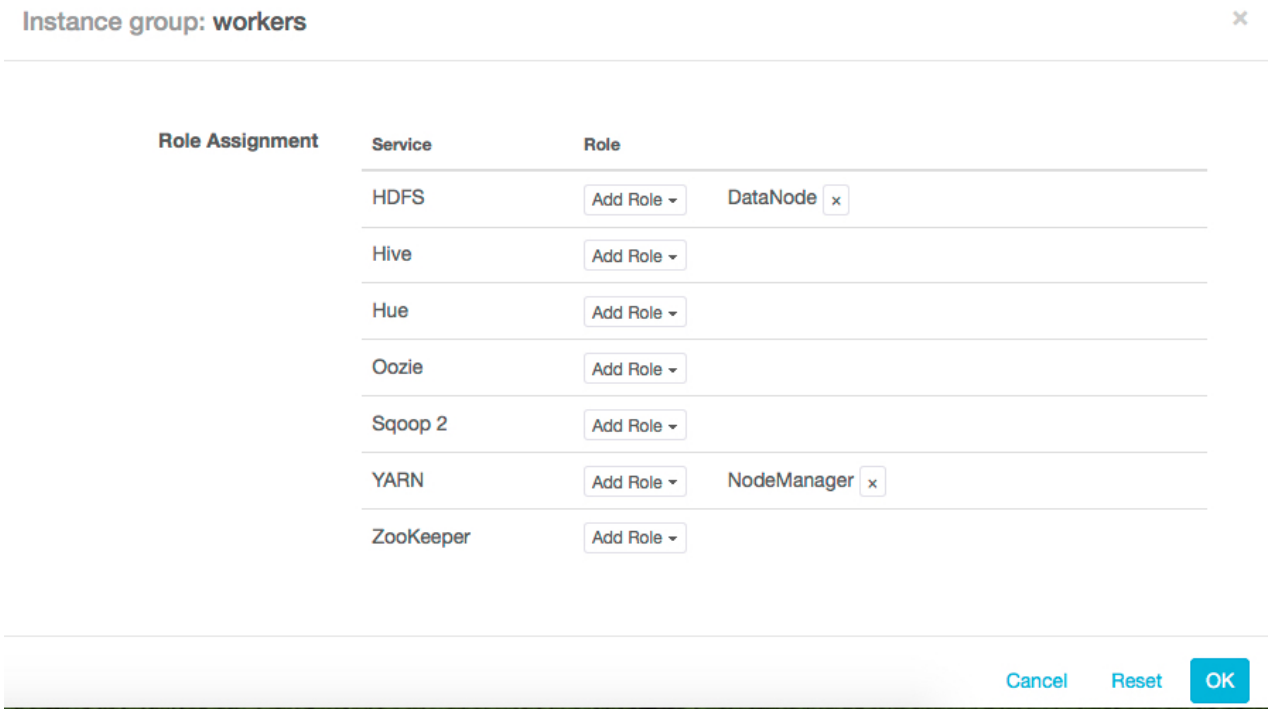
For an introduction to master, worker, and gateway roles, see the Cloudera Manager 5 Overview .

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:



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:



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 185.

Ensure Consistency of Instances before Modify Operations

Operations that modify a cluster, such as grow and clone operations, require all the instances in an instance group to be consistent, that is, to have the same role assignments and configurations in Cloudera Manager, and the same properties, such as instance type and image, in the cloud provider.

With Cloudera Director 2.5 and higher, the cluster refresher will detect changes to the instances in Cloudera Manager and in the cloud provider. If the instances in an instance group are inconsistent, the UI will flag the inconsistent instances and prevent grow operations on the cluster. During this time, clone operations on the cluster will fail. Using Cloudera Manager or your cloud provider's tools, you must change role assignments or configurations, or change properties in the cloud provider, such as instance type or image, to make the instances in your instance groups consistent. Once all the instances in the virtual instance group are consistent, the instance template will be updated, and subsequent grow operations on the group, or clone operations on the cluster, will use the new role assignments and configurations, and the new cloud provider properties.

If you are unable to make the instances consistent, you can use the following workarounds. If you created your cluster from a configuration file, you can edit the configuration file before using it to create a new cluster. And if you created the cluster via the UI, you can use the export functionality to get a configuration file, and edit that configuration file to create a new cluster with consistency of instances within instance groups.

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 185.

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 169.

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 Commands on page 12.

Diagnostic Data Collection

Cloudera Manager log files provide important information for Cloudera Support to use in analyzing problems or unexpected behavior with Cloudera Manager deployments or CDH clusters. Cloudera Director triggers the collection of diagnostic data for deployments and clusters it manages. This helps prevent situations where a failed cluster has been terminated but Cloudera Support has no diagnostic data or log files to help identify the cause of the failure. If you have a Cloudera Enterprise or Cloudera Enterprise Trial license, diagnostic data is collected and sent to Cloudera Support automatically on cluster bootstrap or update failure. By default, diagnostic data is also downloaded to the Cloudera Director instance.

If Cloudera Manager cannot collect diagnostic data, no information is sent to Cloudera Support, and the Cloudera Manager service logs are downloaded to Cloudera Director instead of the diagnostic data. The logs contain less information than the diagnostic data, but can still be useful to Cloudera Support for analyzing deployment and cluster behavior.

Note: If you are using a Cloudera Express license instead of a Cloudera Enterprise license, the **Collect Diagnostic Data** action results in the downloading of Cloudera Manager service logs to Cloudera Director. These logs are not uploaded to Cloudera Support.

You can initiate diagnostic data collection manually through the Cloudera Director web UI or API. You can collect diagnostic data for an entire Cloudera Manager deployment or for a specific CDH cluster.

For more information on how diagnostic data collection works in Cloudera Manager, see the Cloudera Manager documentation page Sending Usage and Diagnostic Data to Cloudera.

Manual Collection of Diagnostic Data

You can manually trigger the collection of diagnostic data using either the Cloudera Director web UI or the Cloudera Director API.

Using the Web UI

To trigger diagnostic data collection, perform the following steps:

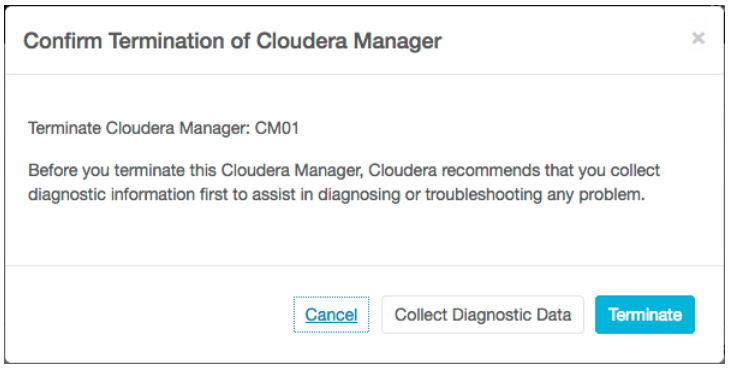
1. Go to the Cloudera Director web UI page for the deployment or cluster.
2. Click the down arrow on the dropdown list to the right of the deployment or cluster name.
3. In the dropdown list, click **Collect Diagnostic Data**.

Cloudera Director makes an API call to the Cloudera Manager API `collectDiagnosticData`. If successful, Cloudera Manager sends the diagnostic data to Cloudera Support and, if the **download diagnostic data** property is set to `true` in the Cloudera Director `application.properties` file, also downloads a zip file containing the diagnostic data for the deployment or cluster to the Cloudera Director EC2 instance. If diagnostic data collection is unsuccessful, and the

download diagnostic data property is set to `true`, Cloudera Manager downloads the Cloudera Manager service logs to Cloudera Director.

Manually Triggering Collection of Diagnostic Data at Cluster Termination

When you terminate a Cloudera Manager deployment or CDH cluster in the web UI, the screen for confirming the termination includes a button that triggers collection of diagnostic data:



Note: Diagnostic data collection is also triggered before termination when you invoke the `terminate-remote` command with the Cloudera Director CLI. There is no separate CLI command to trigger collection of diagnostic data, so you must use the web UI or API to trigger diagnostic data collection without terminating the deployment or cluster.

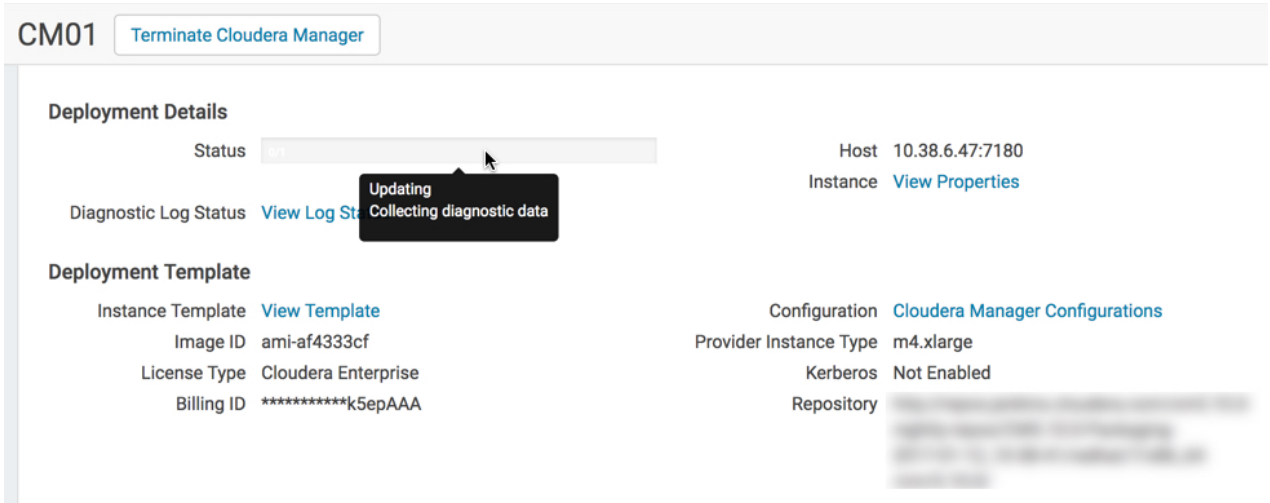
Using the API

To manually trigger collection of diagnostic data for Cloudera Manager deployments, use the API at `http://cloudera_director_ip:port_number/api-console/index.html#!/deployments/collectDiagnosticData`.

To manually trigger collection of diagnostic data for CDH clusters, use the API at `http://cloudera_director_ip:port_number/api-console/index.html#!/clusters/collectDiagnosticData`.

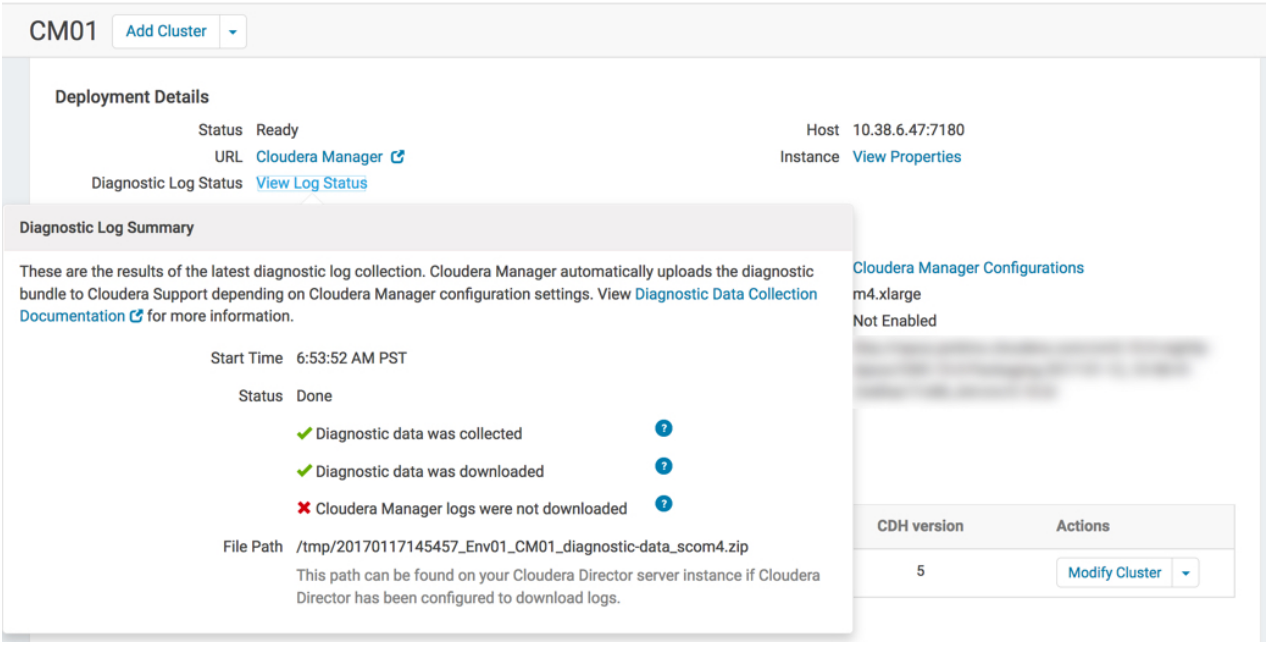
Status for Data Collection

While diagnostic data collection is in progress, the status of the deployment or cluster changes from its current state to **Updating: Collecting diagnostic data** when you mouse-over the **Status** bar:



The cluster status is not actually updated; the updating message is displayed simply to inform that diagnostic data collection is in progress. Because diagnostic data collection does not change the status of the cluster, when the data collection is complete, the deployment or cluster status message reverts to what it was before diagnostic data collection began.

If you click **View Log Status** on the deployment or cluster screen, the **Diagnostic Log Summary** is displayed, showing information about the last diagnostic data collection:



If diagnostic data has never been collected for the deployment or cluster, the **Diagnostic Log Status** value is `Not Collected` and there is no link to open the **Diagnostic Log Status** screen.

Configuring Diagnostic Data Collection

By default, Cloudera Manager sends diagnostic data to Cloudera Support and to Cloudera Director. You can configure diagnostic data collection on Cloudera Manager and Cloudera Director using the procedures described in this section.

Configuring Upload of Diagnostic Data to Cloudera Support

The Cloudera Manager server property that determines whether diagnostic data is automatically sent to Cloudera Support has the display name **Send Diagnostic Data to Cloudera Automatically** and the API name `phone_home`. The default value for this property is `true`. To disable diagnostic data collection in Cloudera Manager, set this property to `false`. Set the property in Cloudera Manager by following these steps:

1. In Cloudera Manager, click **Administration** > **Settings**.
2. In the list of **Filters** in the lefthand pane, click **Support**.
3. Click the checkbox for the property **Send Diagnostic Data to Cloudera Automatically** to toggle the setting between `true` and `false`.

For more information on the `phone_home` property, see the table in the **Support** section of Cloudera Manager Server Properties.

Configuring Download of Diagnostic Data to Cloudera Director

Several Cloudera Director server configuration properties affect the way diagnostic data is handled. You can set these properties in the `application.properties` file located at `/etc/cloudera-director-server/` on the Cloudera Director instance, or at the command line.

- `lp.debug.collectDiagnosticDataOnFailure`: Determines whether automatic collection of diagnostic data occurs for cluster bootstrap or update failures. The default value is `true`.
- `lp.debug.downloadDiagnosticData`: Determines whether diagnostic data is downloaded to the Cloudera Director instance. The default value is `true`.
- `lp.remote.terminate.assumeYes`: Determines whether Cloudera Director skips prompting the user to confirm termination when the `terminate-remote` command is invoked. If you set the property to `true`, termination proceeds even if diagnostic data collection has failed. The default setting is `false`.

- `lp.debug.diagnosticDataDownloadDirectory`: Sets a nondefault path for the download of diagnostic data for deployments and clusters. The default location is `/tmp`. The directory where diagnostic data has been downloaded appears in the **File Path** field in the **Diagnostic Log Summary**.
- `lp.debug.createDiagnosticDataDownloadDirectory`: Determines whether Cloudera Director creates the nondefault download directory specified in `lp.debug.diagnosticDataDownloadDirectory` if it does not exist. The default value is `true`.

For information about setting Cloudera Director properties by using the CLI or editing the `application.properties` file, see Setting Cloudera Director Properties on page 153.

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**.
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/v9/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/v9/users`	Lists all usernames.
`POST /api/v9/users`	Creates a new user account (Admin role required).
`GET /api/v9/users/current`	Gets account information on the currently logged-in user.
`GET /api/v9/users/{username}`	Gets account information on a user.
`PUT /api/v9/users/{username}`	Changes account information on a user.
`DELETE /api/v9/users/{username}`	Deletes an account (Admin role required)
`PUT /api/v9/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 45.
- For Google Cloud Platform, see Installing Cloudera Director Server and Client on Google Compute Engine on page 61.

Important: Cloudera Director requires a JDK. For more information, see Supported Software and Distributions on page 37.

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.

   Note: Spark 2.2 requires JDK 8 and Python 2.7 or higher. For information on using Java 8 with Cloudera Director, see Deploying a Java 8 Cluster.

   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. The Cloudera Director configuration file is in HOCON format. For information on HOCON, see the documentation at https://github.com/typesafehub/config/blob/master/HOCON.md.

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.

   With Cloudera Director 2.2 and later, you can use the `validate` CLI command to check your configuration file's settings for environments, deployments, and clusters. For more information, see the entry for `validate` on the page Cloudera Director Interfaces.

> **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.

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@director-client-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 135.

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

> **Note:** See Requirements and Supported Versions for the latest information about operating system versions supported by versions of 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. Note that the Cloudera Manager minor version must the same as or higher than the CDH minor version. For instance, Cloudera Manager 5.7 can be used to launch and manage a CDH 5.5 cluster but not with a CDH 5.10. 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 and Higher

- Cloudera Director 2.0 and higher 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.

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.

> **Note:** Cloudera Director 2.5 will fail to start after upgrading from 2.4 if a cluster contains a heterogeneous instance group. To resolve this problem, find heterogeneous instance groups displayed in the Cloudera Director UI. Resolve any inconsistencies within each instance group to restore homogeneity before upgrading Cloudera Director from 2.4 to 2.5. For more information about the consistency of virtual instance groups, see Ensuring Consistency of Virtual Instance Groups.

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 and higher require 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`, `/2.2.0`, or `/2.2`. 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 and higher require 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-director2.2`, or `trusty-director2.2.0`). 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 198. 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 117. For more information on AWS IAM, see the IAM User Guide in the AWS documentation.

Troubleshooting Cloudera Director

This topic contains information on problems that can occur when you set up, configure, or use Cloudera Director, their causes, and their solutions.

Viewing Cloudera Director Logs

To help you troubleshoot problems, you can view the Cloudera Director logs. Log files are 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
```

Cloudera Director normally logs only error, warning, and informational messages. To configure it to log debug level messages, edit the file `logback.xml`, which can be found at the following locations:

- Cloudera Director client: `/etc/cloudera-director-client/logback.xml`
- Cloudera Director server: `/etc/cloudera-director-server/logback.xml`

The XML file configures the logback logging library. To turn on all debug logging for Cloudera Director and its libraries, change the "root" element as follows:

```
<root level="DEBUG">
```

Enabling debug logging significantly increases the size of the logs, and may include more information than needed for troubleshooting. Once you discover specific loggers that carry information you care most about, you can narrow the scope of debug logging to those only. For example, if after turning on all debug logging you find that the messages emitted from Cloudera Director itself are most important, then you can set the root level back to `INFO` and then add a new `logger` element like this example, along with the other similar elements.

```
<logger name="com.cloudera.launchpad" level="DEBUG"/>
```

The `logback.xml` file can be reconfigured in many other ways to adjust how logging is performed. See Logback configuration in the Logback project documentation to learn more. Note that major changes to log format and contents will hamper the effectiveness of Cloudera support, if you should need to forward logs to them as part of troubleshooting.

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, use the H2 backup tools instead of simply copying the file. For more information, see the H2 Tutorial.

Manual Modifications to the Cloudera Director Database

Manual modifications to the Cloudera Director database *are not supported*. Modifications made outside of Cloudera Director control can lead to permanent data loss and unrecoverable errors in Cloudera Director.

Slow or Failed OS Updates in Some AWS Regions

Symptom

In AWS, Cloudera Director triggers operating system updates and performs software downloads on instances it allocates in your chosen region. Depending on the local network configuration, these updates and downloads may go slowly or fail.

Solutions

Consider trying one or more of the following steps:

- **Disable instance normalization.** This causes Cloudera Director to not perform usual automated, general work on new instances. You should replace that work with your own, either by building a custom AMI with the work already accomplished, or by using a bootstrap script. Normalization can be disabled using a configuration file; contact Cloudera support for guidance on what else you need to do.
- **Create a preloaded AMI.** Cloudera Director can avoid downloading Cloudera Manager and CDH software if it is already present in expected locations on instances. This also speeds up deployment and cluster bootstrap processes, even when download speeds from Cloudera repositories are reasonable. See the documentation for more information.
- **Mirror Cloudera repositories.** Instead of preloading an AMI with Cloudera software, you can host them at local mirrors, and point Cloudera Director to them as alternative download locations. As with preloaded AMIs, taking this step can speed up bootstrap processes, and make your architecture less vulnerable to network problems. See the documentation for more information.

Cloudera Manager API Call Fails

Symptom

A Cloudera Manager API call fails in Cloudera Director.

Cause

Needs to be diagnosed. (See **Solution** immediately below.)

Solution

Enable API debugging in Cloudera Manager by going to **Settings** on the **Administration** tab in Cloudera Manager and clicking the checkbox **Enable Debugging of API**. Then look at the Cloudera Manager server logs to get more information on why the API call failed.

Note: You can also enable Cloudera Manager API debugging in the configuration file when launching a cluster by setting `enable_api_debug: true` in the Cloudera Manager configs section. The sample configuration file aws.reference.conf has this set by default.

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 unable to manage the cluster. See Cloudera Director and Cloudera Manager Usage on page 173.

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.

RDS Name Conflicts

Symptom

RDS name conflicts occur when creating multiple clusters with the same configuration file.

Cause

Most often, deletion of an older RDS instance has not completed when you try to launch a new cluster using the same configuration file, and therefore the same RDS name.

Solution

Allow more time for an RDS instance to be completely removed before creating a new cluster with the same configuration file, or change the name of the RDS instance in the configuration files for new clusters.

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 required roles are assigned when provisioning a cluster. This can lead to failures during the intial 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 fails 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

Cloudera Director fails to bootstrap a cluster with a DNS error. The Cloudera Agent log (`cloudera-scm-agent.log`) will show an entry similar to the following:

```
[27/Mar/2017 20:26:16 +0000] 12596 Thread-13 https ERROR Failed to retrieve/stroe URL:
http://ip-10-202-202-109.ec2.internal:7180/cmf/parcel/download/CDH-5.10.0-1.cdh5.10.0.p0.41-el7.parcel.torrent
  ->
/opt/cloudera/parcel-cache/CDH-5.10.0-1.cdh5.10.0.p0.41-el7.parcel.torrent
<urlopen error [Errno -2] Name or service not known>
```

Cause

This can be caused by one the following:

- **DNS Hostnames** is not set to **Yes** in the **Edit DNS Hostnames** VPC configuration setting.
- The Amazon Virtual Private Cloud (VPC) is not set up for forward and reverse hostname resolution. Forward and reverse DNS resolution is a 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** is displayed.
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 is displayed.
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 are using AmazonProvidedDNS in `us-east-1`, specify `ec2.internal`. If you are 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 1 GB of free memory. See Resource Requirements on page 38 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 184 (Note) 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 41. For commands to turn off iptables, see either Installing Cloudera Director Server and Client on the EC2 Instance on page 45 or Installing Cloudera Director Server and Client on Google Compute Engine on page 61. 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

Can I move Cloudera Director to a different instance?

In some circumstances, you may want to move Cloudera Director from the instance where it is installed to a different instance, for example, if you need to delete the AWS subnet where the Cloudera Director instance is currently running. This can be done much as it would be with any other application.

The procedure is to snapshot or image the current host and deploy a new instance from it in the new subnet. Cloudera recommends that you take a backup of the database before the migration to be safe. If you are using the H2 embedded database (which is *not* recommended for production deployments), it resides on the file system, so a snapshot or image will include it. MySQL will reside elsewhere on a database server. Backing it up in either case is an extra precaution, but isn't strictly necessary in order to move Cloudera Director over to the new instance.

Note: If you are using MySQL for the Cloudera Director database, the security rules need to allow instances in the new subnet to reach the MySQL server, whether in RDS or elsewhere.

Can master or worker roles be run on instances where Cloudera Manager is running?

No, CDH cluster entities cannot be run on the same instance as Cloudera Manager.

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 Using Custom Repositories with Cloudera Manager and CDH on page 107.

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 167.

How do I create instances in multiple availability zones in AWS EC2?

This is AWS-specific. Each subnet exists in only one availability zone, so if you want multiple availability zones for your instances, you need to create multiple instance groups, with each one having a template that points to a different subnet.

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.

transient cluster

A short lived cluster that launches, processes a set of data, and terminates. Transient 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, 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 is 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.

Appendix: Apache License, Version 2.0

SPDX short identifier: Apache-2.0

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims

licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

```
Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

  http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```