

cloudera[®]

Cloudera Altus Director User Guide

Important Notice

© 2010-2021 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 Altus Director 6.0.x
Date: May 20, 2021

Table of Contents

Introduction.....	10
Altus Director Features.....	10
Displaying Altus Director Documentation.....	11
Altus Director Interfaces.....	11
<i>Web User Interface.....</i>	<i>11</i>
<i>Command Line Interface.....</i>	<i>12</i>
<i>API.....</i>	<i>13</i>
<i>Altus Director Interface Usage.....</i>	<i>13</i>
Altus Director Release Notes.....	15
New Features and Changes in Cloudera Altus Director.....	15
<i>What's New in Altus Director 6.0.....</i>	<i>15</i>
<i>Changes in Altus Director 6.0.....</i>	<i>16</i>
Known Issues and Workarounds in Altus Director.....	19
<i>Hue service in CDH 6 that uses the wrong version of Python fails to start</i>	<i>19</i>
<i>Incorrect migration of Cloudera Altus Director's database from H2 to MySQL causes upgrade to 2.8 to fail.....</i>	<i>19</i>
<i>Added instance groups are not configured based on their instance type.....</i>	<i>19</i>
<i>Edited or cloned Altus Director 2.8 instance templates for Azure do not work as expected.....</i>	<i>20</i>
<i>Environment deletion fails with a 500 response code.....</i>	<i>20</i>
<i>Maximum length of environment name is misleading.....</i>	<i>20</i>
<i>AWS rate limiting due to large number of EBS volumes.....</i>	<i>20</i>
<i>Altus Director cannot deploy Cloudera Navigator Key Trustee Server.....</i>	<i>20</i>
<i>Changes to Cloudera Manager username and password must also be made in Altus Director.....</i>	<i>20</i>
<i>Altus Director might use AWS credentials from instance of Altus Director server.....</i>	<i>21</i>
<i>Root partition resize fails on CentOS 6.5 (HVM).....</i>	<i>21</i>
<i>When using RDS and MySQL, Hive Metastore canary can fail in Cloudera Manager.....</i>	<i>21</i>
Issues Fixed in Altus Director.....	21
<i>Issues Fixed in Altus Director 6.0.1.....</i>	<i>21</i>
<i>Issues Fixed in Altus Director 6.0.0.....</i>	<i>22</i>
Unsupported Components and Features in Altus Director.....	23
Requirements and Supported Versions.....	24
Cloud Providers.....	24
Altus Director Service Provider Interface (SPI).....	24
Supported Software and Distributions.....	24
Resource Requirements.....	26

Supported Cloudera Manager and CDH Versions.....	27
Cloudera Data Science Workbench Support.....	27
Networking and Security Requirements.....	27
Supported Browsers.....	28

Getting Started with Altus Director.....29

Getting Started on Amazon Web Services (AWS).....	29
<i>Preparing Your AWS EC2 Resources.....</i>	<i>29</i>
<i>Launching an EC2 Instance for Altus Director.....</i>	<i>30</i>
<i>Installing Altus Director Server and Client on the EC2 Instance.....</i>	<i>34</i>
<i>Configuring a SOCKS Proxy for Amazon EC2.....</i>	<i>37</i>
<i>Adding a Altus Director Environment on AWS.....</i>	<i>39</i>
<i>Simple Setup: Creating a Cluster on AWS with Default Settings.....</i>	<i>41</i>
<i>Advanced Setup: Installing Cloudera Manager and CDH on AWS.....</i>	<i>43</i>
<i>Pausing a Cluster on AWS.....</i>	<i>48</i>
<i>Cleaning Up Your AWS Deployment.....</i>	<i>49</i>
Getting Started on Google Cloud Platform.....	50
<i>Creating a Google Cloud Platform Project.....</i>	<i>50</i>
<i>Configuring Tools for Your Google Cloud Platform Account.....</i>	<i>50</i>
<i>Creating a Google Compute Engine VM Instance.....</i>	<i>52</i>
<i>Installing Altus Director Server and Client on Google Compute Engine.....</i>	<i>53</i>
<i>Configuring a SOCKS Proxy for Google Compute Engine.....</i>	<i>56</i>
<i>Deploying Cloudera Manager and CDH on Google Compute Engine.....</i>	<i>56</i>
<i>Cleaning Up Your Google Cloud Platform Deployment.....</i>	<i>61</i>
Getting Started on Microsoft Azure.....	61
<i>Obtaining Credentials for Altus Director.....</i>	<i>61</i>
<i>Setting up Azure Resources.....</i>	<i>62</i>
<i>Setting Up Dynamic DNS on Azure.....</i>	<i>67</i>
<i>Setting Up MySQL or PostgreSQL.....</i>	<i>74</i>
<i>Setting Up a Virtual Machine for Altus Director Server.....</i>	<i>75</i>
<i>Installing Altus Director Server and Client on Azure.....</i>	<i>75</i>
<i>Configuring a SOCKS Proxy for Microsoft Azure.....</i>	<i>77</i>
<i>Allowing Access to VM Images.....</i>	<i>78</i>
<i>Adding an Altus Director Environment on Azure.....</i>	<i>78</i>
<i>Simple Setup: Creating an Azure Cluster with Default Settings.....</i>	<i>80</i>
<i>Advanced Setup: Installing Cloudera Manager and CDH on Azure.....</i>	<i>82</i>
<i>Terminating an Azure Deployment.....</i>	<i>86</i>
<i>Adding New VM Images, Custom VM Images, Regions, and Instances.....</i>	<i>86</i>
<i>Important Notes About Altus Director and Azure.....</i>	<i>87</i>

Usage-Based Billing.....93

Prerequisites.....	93
--------------------	----

How Usage-Based Billing Works.....	93
Deploying Cloudera Manager and CDH with Usage-Based Billing.....	94
<i>Enabling Usage-Based Billing with the Altus Director Server web UI.....</i>	<i>94</i>
<i>Enabling Usage-Based Billing with bootstrap-remote.....</i>	<i>94</i>
Managing Billing IDs with an Existing Deployment.....	95
Troubleshooting Network Connectivity for Usage-Based Billing.....	95
Altus Director Usage Bundles.....	96
<i>Metadata Section.....</i>	<i>96</i>
<i>Cloudera Manager Block.....</i>	<i>96</i>
<i>Altus Director Block.....</i>	<i>97</i>
<i>Usage Logging.....</i>	<i>97</i>

Using Cloud Provider Regions.....99

Running Altus Director and Cloudera Manager in Different Regions or Clouds.....	99
Using a New AWS Region in Altus Director.....	100
<i>Entering the Region Code.....</i>	<i>100</i>
<i>Region Endpoints.....</i>	<i>100</i>
<i>Other Considerations.....</i>	<i>101</i>

Configuring Storage for Altus Director.....102

Using MySQL for Altus Director Server.....	102
<i>Installing the MySQL Server.....</i>	<i>102</i>
<i>Configuring and Starting the MySQL Server.....</i>	<i>102</i>
<i>Installing the MySQL JDBC Driver.....</i>	<i>104</i>
<i>Creating a Database for Altus Director Server.....</i>	<i>105</i>
<i>Configuring Altus Director Server to use the MySQL Database.....</i>	<i>106</i>
<i>Migrating Altus Director Server from H2 to MySQL.....</i>	<i>106</i>
<i>Step 1: Prepare databases.....</i>	<i>106</i>
<i>Step 2: Export data from H2 database.....</i>	<i>107</i>
<i>Step 3: Prepare MySQL database for data import.....</i>	<i>114</i>
<i>Step 4: Import data to MySQL.....</i>	<i>115</i>
Using MariaDB for Altus Director Server.....	115
<i>Installing the MariaDB Server.....</i>	<i>115</i>
<i>Configuring and Starting the MariaDB Server.....</i>	<i>116</i>
<i>Installing the MariaDB JDBC Driver.....</i>	<i>117</i>
<i>Creating a Database for Altus Director Server.....</i>	<i>118</i>
<i>Configuring Altus Director Server to use the MariaDB Database.....</i>	<i>118</i>
Altus Director Database Encryption.....	119
<i>Cipher Configuration.....</i>	<i>119</i>
<i>Starting with Encryption.....</i>	<i>120</i>
<i>Changing Encryption.....</i>	<i>121</i>
Shrinking an H2 Database.....	122

Configuring Storage for Cloudera Manager and CDH.....124

Using an External Database for Cloudera Manager and CDH.....	124
<i>Defining External Database Servers.....</i>	<i>124</i>
<i>Defining External Databases.....</i>	<i>130</i>
Using EBS Volumes for Cloudera Manager and CDH.....	133
<i>EBS Volume Types.....</i>	<i>133</i>
<i>Configuring EBS Volumes.....</i>	<i>134</i>
Using Amazon S3 Object Storage.....	137
<i>Configuring Amazon S3 with Altus Director.....</i>	<i>137</i>

Security, Encryption, and High Availability.....141

Enabling TLS with Altus Director.....	141
<i>Automatic TLS (Auto-TLS).....</i>	<i>141</i>
<i>Manual TLS.....</i>	<i>146</i>
<i>Working with an Existing Cloudera Manager.....</i>	<i>148</i>
<i>TLS for Altus Director Itself.....</i>	<i>148</i>
<i>TLS for Altus Director's Database.....</i>	<i>150</i>
<i>Using Custom DNS with Auto-TLS on AWS.....</i>	<i>152</i>
SSH Host Key Retrieval and Verification.....	154
<i>NONE.....</i>	<i>154</i>
<i>PROVIDER.....</i>	<i>154</i>
<i>INSTANCE.....</i>	<i>155</i>
<i>FALLBACK.....</i>	<i>155</i>
Creating Kerberized Clusters With Altus Director.....	155
<i>Creating a Kerberized Cluster with the Altus Director Configuration File.....</i>	<i>155</i>
Enabling Sentry Service Authorization.....	157
<i>Prerequisites.....</i>	<i>157</i>
<i>Setting Up the Sentry Service Using the Altus Director CLI.....</i>	<i>157</i>
<i>Setting up the Sentry Service Using the Altus Director API.....</i>	<i>158</i>
<i>Related Links.....</i>	<i>158</i>
Creating Highly Available Clusters With Altus Director.....	158
<i>Limitations and Restrictions.....</i>	<i>159</i>
<i>Editing the Configuration File to Launch a Highly Available Cluster.....</i>	<i>159</i>
<i>Migrating HDFS Master Roles.....</i>	<i>161</i>
Encrypted Configuration Properties.....	164
<i>Creating an Encrypted Configuration Property.....</i>	<i>164</i>
<i>Using Encrypted Configuration Properties.....</i>	<i>164</i>
<i>Configuring Encryption of Configuration Properties.....</i>	<i>164</i>

Configuring and Running Altus Director.....166

Auto-Repair for Failed or Terminated Instances.....	166
---	-----

Deploying Java on Cluster Instances.....	169
<i>AUTO JDK Installation Strategy.....</i>	<i>169</i>
<i>DIRECTOR_MANAGED JDK Installation Strategy.....</i>	<i>170</i>
<i>NONE JDK Installation Strategy.....</i>	<i>170</i>
Setting Altus Director Properties.....	171
Pausing Altus Director Instances.....	172
<i>Pausing the Altus Director Server.....</i>	<i>172</i>
<i>Pausing a Cluster in Azure.....</i>	<i>172</i>
Configuring Altus Director Server for LDAP and Active Directory.....	173
<i>User and Group Model.....</i>	<i>173</i>
<i>Basic LDAP Configuration.....</i>	<i>173</i>
<i>Active Directory Configuration.....</i>	<i>175</i>
Configuring Altus Director for a New AWS Instance Type.....	176
<i>Updated Virtualization Mappings.....</i>	<i>176</i>
<i>Updated Ephemeral Device Mappings.....</i>	<i>177</i>
<i>Using the New Mappings.....</i>	<i>177</i>
Configuring Altus Director to Use Custom Tag Names on AWS.....	178
<i>Using the New Mappings.....</i>	<i>178</i>

Using Altus Director Server to Manage Cloudera Manager Instances.....179

Altus Director and Cloudera Manager Usage.....	179
<i>When to Use Altus Director.....</i>	<i>179</i>
<i>When to Use Cloudera Manager.....</i>	<i>179</i>
<i>CDH Cluster Management Tasks.....</i>	<i>180</i>
<i>Ensuring Consistency of Virtual Instance Groups.....</i>	<i>182</i>
<i>CDH Cluster Management Guidelines for Altus Director.....</i>	<i>183</i>
Setting Cloudera Manager Configurations.....	183
<i>Cluster Configuration Using Cloudera Manager.....</i>	<i>183</i>
<i>Setting up a Cloudera Manager License.....</i>	<i>184</i>
<i>Deployment Template Configuration.....</i>	<i>184</i>
<i>Cluster Template Service-wide Configuration.....</i>	<i>185</i>
<i>Cluster Template Roletype Configurations.....</i>	<i>186</i>
Ports Used by Altus Director.....	187
SSH Keys in Altus Director.....	188
<i>The Role of Keys in SSH.....</i>	<i>188</i>
<i>Use of SSH Keys in Cloud Providers.....</i>	<i>188</i>
<i>Altus Director's Use of SSH Keys.....</i>	<i>189</i>
<i>Good Practices for SSH Key Management with Altus Director.....</i>	<i>189</i>
Creating AWS Identity and Access Management (IAM) Policies.....	189
Using Custom Repositories with Cloudera Manager and CDH.....	192
<i>Creating a Custom Repository.....</i>	<i>192</i>
<i>Creating a Cloudera Manager and CDH AMI.....</i>	<i>192</i>
Cloudera Manager Health Information.....	192

Opening Cloudera Manager.....	193
Diagnostic Data Collection.....	193
<i>Manual Collection of Diagnostic Data.....</i>	<i>193</i>
<i>Configuring Diagnostic Data Collection.....</i>	<i>195</i>
User Management.....	196
<i>Managing Users with the Altus Director Web web UI.....</i>	<i>196</i>
<i>Managing Users with the Altus Director API.....</i>	<i>197</i>

Using Altus Director Server to Manage Cluster Instances.....199

The Altus Director Configuration File.....	199
<i>Location of Sample Configuration Files.....</i>	<i>199</i>
<i>Customizing the Configuration File.....</i>	<i>199</i>
<i>Valid Role Types for Use in Configuration Files.....</i>	<i>199</i>
<i>Using the API to Import a Configuration File.....</i>	<i>199</i>
<i>Default Values for Configuration Files.....</i>	<i>200</i>
Submitting a Cluster Configuration File.....	200
Exporting a Configuration File.....	201
Deploying Clusters in an Existing Environment.....	201
Using Spot Instances.....	202
<i>Planning for Spot Instances.....</i>	<i>203</i>
<i>Specifying Spot Instances.....</i>	<i>203</i>
<i>Spot Blocks: Specifying a Duration for Spot Instances.....</i>	<i>204</i>
<i>Best Practices for Using Spot Instances.....</i>	<i>205</i>
Using Products outside CDH with Altus Director.....	205
<i>Custom Service Descriptors.....</i>	<i>205</i>
<i>Using Kudu with CDH 5.12 or Earlier.....</i>	<i>206</i>
<i>Using Spark 2 with Altus Director.....</i>	<i>207</i>
<i>Using Cloudera Data Science Workbench with Altus Director.....</i>	<i>208</i>
<i>Using Third-Party Products with Altus Director.....</i>	<i>209</i>
Creating and Modifying Clusters with the Altus Director Web UI.....	210
<i>Configuring Instance Groups During Cluster Creation.....</i>	<i>210</i>
<i>Modifying the Number of Instances in an Existing Cluster.....</i>	<i>211</i>
<i>Repairing Worker and Gateway Instances in a Cluster.....</i>	<i>213</i>
Post-Creation Scripts.....	214
<i>Deployment Post-creation Scripts.....</i>	<i>214</i>
<i>Cluster Post-creation Scripts.....</i>	<i>215</i>
<i>Predefined Environment Variables</i>	<i>217</i>
Terminating a Cluster.....	217
<i>Terminating a Cluster with the web UI.....</i>	<i>217</i>
<i>Terminating a Cluster with the CLI.....</i>	<i>218</i>

Using the Altus Director Client.....219

Installing Altus Director Client.....	219
Provisioning a Cluster on AWS.....	220
Running Altus Director Client.....	222

Altus Director Upgrade.....223

Step 1: Review the Upgrade Requirements.....	223
Step 2. Perform Tasks Required for the Upgrade.....	224
Step 3. Upgrade to Altus Director 6.0.....	225
<i>RHEL and CentOS</i>	225
<i>Ubuntu</i>	226
Step 4. Restore the Configuration	226
Step 5. Restart Director.....	227

Troubleshooting Altus Director.....228

Bootstrap fails in Azure when custom image has an attached data disk and dataDiskCount is not 0.....	229
Slow or Failed OS Updates in Some AWS Regions.....	229
Altus Director Bootstrap Fails with DNS Error.....	230
Altus Director Bootstrap Fails with IAM Permissions Error.....	230
Cloudera Manager API Call Fails.....	230
Altus Director Cannot Manage a Cluster That Was Kerberized Through Cloudera Manager.....	231
RDS Name Conflicts.....	231
New Cluster Fails to Start Because of Missing Roles.....	231
Altus Director Server Will Not Start with Unsupported Java Version.....	232
Error Occurs if Tags Contain Unquoted Special Characters.....	232
DNS Issues.....	232
Server Does Not Start.....	233
Problem When Removing Hosts from a Cluster.....	234
Problems Connecting to Altus Director Server.....	234

Frequently Asked Questions.....235

General Questions.....	235
------------------------	-----

Altus Director Glossary.....237

Appendix: Apache License, Version 2.0.....239

Introduction

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

Altus 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 Altus Director programmatically through the REST API or the CLI to maximize time-to-value for an enterprise data hub in cloud environments.

Altus 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](#).

Altus Director Features

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

Benefit	Features
Simplified cluster lifecycle management	Simple user interface: <ul style="list-style-type: none"> • Self-Service spin up and tear down • Scaling of clusters for spiky workloads • Simple cloning of clusters • Cloud blueprints for repeatable deployments
Elimination of lock-in	Flexible, open platform: <ul style="list-style-type: none"> • 100% open source Hadoop distribution • Native support for hybrid deployments • Third-party software deployment in the same workflow • Support for custom, workload-specific deployments
Accelerated time to value	Enterprise-ready security and administration: <ul style="list-style-type: none"> • Support for complex cluster topologies • Minimum size cluster when capacity constrained • Management tooling • Compliance-ready security and governance • Backup and disaster recovery with an optimized cloud storage connector
Reduced support costs	Monitoring and metering tools:

Benefit	Features
	<ul style="list-style-type: none"> • Multi-cluster health dashboard • Instance tracking for account billing

Displaying Altus Director Documentation

To display Altus 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 Altus 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 Altus Director server is running, you must restart the server in order for your changes to take effect:

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

Altus Director Interfaces

Altus Director provides different user interfaces for centralized deployment, configuration, and administration of Cloudera Manager and CDH clusters in the cloud. After you complete the Altus 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.

Altus 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 Altus Director API to programmatically control the lifecycle of your clusters. Cloudera provides SDKs for the Python and Java programming languages.

Web User Interface

After you install the Altus Director server, you can use a browser to access the Altus 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, Altus Director saves the state of the cluster in the Altus Director database. The database can store deployment information about multiple environments, deployments, and clusters that are deployed and managed by Altus Director. The deployment information in the database allows Altus Director to create additional clusters in the managed deployments.

By default, Altus Director saves deployment information in an H2 database, but the H2 database is only supported for proof-of-concept clusters or for testing. Configure Altus Director to use an externally managed MySQL or MariaDB database for production clusters. Specify an external database in the `application.properties` file in the server host. For more information, see [Configuring Storage for Altus Director](#) on page 102.

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 most configuration and administrative tasks on a Cloudera Manager deployment. When you perform a complex or customized deployment or configuration, you might have to use the command line interface with a configuration file or to use the API. For example, you cannot update the TLS settings of a deployment on the web UI. You must use the CLI or the API to enable or disable TLS in a deployment.

Command Line Interface

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

When you use the command line interface to deploy a cluster, the state of the cluster is saved in the Altus Director database. The database can store deployment information about multiple environments, deployments, and clusters that are deployed and managed by Altus Director. The deployment information in the database allows Altus 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 Altus Director client, the installation creates a configuration file named `application.properties`. The properties file includes configuration properties such as the Altus 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 Altus 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. Altus 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 Altus 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 [Altus Director scripts GitHub page](#).

Commands

The command line interface includes the following commands:

Command	Description
<code>bootstrap-remote</code>	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 <code>.conf</code> extension. The <code>bootstrap-remote</code> command reads the configuration

Command	Description
	<p>file and creates a cluster with the configuration settings defined in the file. As with Director UI installations, <code>bootstrap-remote</code> speeds up the bootstrap process by configuring Cloudera Manager and the CDH cluster in parallel.</p> <p>To ensure that the command connects to the Altus Director server correctly, you can pass server host and user account properties to the command. The <code>bootstrap-remote</code> command uses the values you pass to connect to the server instead of the values in the <code>application.properties</code> file. For example, you can pass the following properties:</p> <ul style="list-style-type: none"> <code>lp.remote.hostAndPort=host[:port]</code> Hostname and port number of the Altus Director server. The default value in the <code>application.properties</code> file is set to <code>localhost:7189</code>. <code>lp.remote.username=<Altus Director server username></code> Username to use to log in to the Altus Director server. <code>lp.remote.password=<Altus Director server password></code> Password for the Altus Director server user account.
<code>terminate-remote</code>	<p>Terminates a cluster and deployment on a remote server.</p> <p>As in the <code>bootstrap-remote</code> command, you can pass the hostname and port number to connect to the Altus Director server and the username and password to log in to Altus Director.</p>
<code>validate-remote</code>	<p>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.</p> <p>You can set the <code>lp.validate.verbose</code> property to <code>true</code> to output an HTML representation of the configuration.</p>
<code>convert-remote</code>	<p>Converts a simple configuration file to a standard configuration file. For more information, see Converting a Configuration File from Simple to Standard Format on page 42.</p>

API

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

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

Altus Director Interface Usage

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

Task	Web UI	Command line Interface	API
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	■	■	■
Update Cloudera Manager password	■		■
Terminate clusters	■	■	■

Altus Director Release Notes

These release notes provide information on new features and known issues and limitations for Altus Director.

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

New Features and Changes in Cloudera Altus Director

The following sections describe the new features and changes introduced in this release of Altus Director.

What's New in Altus Director 6.0

Altus Director 6.0 provides new features and supports new versions of CDH and Cloudera Manager. It also supports installation of clusters on additional operating system versions.

Cloudera Director Renamed Cloudera Altus Director

Cloudera Director is now part of the Cloudera Altus platform. The full name of Cloudera Director is now Cloudera Altus Director.

The name change does not affect scripts, commands, or file names that use the short name Director or Cloudera.

Support for Cloudera Manager 6.0 and CDH 6.0

By default, Altus Director 6.0 installs Cloudera Manager and CDH 6.0.

Altus Director 6.0 installs Java 8 with Cloudera Manager 6.0 and the CDH 6.0 instances managed by the Cloudera Manager 6.0 instance.

You can use Altus Director to install Cloudera Manager and CDH 5.7 or later versions if you specify the product version and parcel repositories in the Altus Director configuration file. When Altus Director 6.0 installs Cloudera Manager 5.7 or a later version, Altus Director 6.0 installs Java 7 with Cloudera Manager and the CDH instances managed by the Cloudera Manager.

Operating System Support for Altus Director

You can install Altus Director on RHEL/CentOS 7.5 or Ubuntu 16.04.

In addition, Altus Director continues to support the following OS versions:

- CentOS and RHEL, versions 6.7, 6.8, 7.2, 7.4
- Ubuntu 14.04

Support for Java 8

Altus Director 6.0 requires Java 8.

Altus Director 6.0 does not support Java 7 or earlier versions. It also does not support Java 9 or later versions.

Support for Cloudera Manager 5.7 or Higher

Altus Director 6.0 supports Cloudera Manager version 5.7 or higher.

Altus Director 6.0 does not work with Cloudera Manager instances earlier than version 5.7. Altus Director 6.0 will not start if it detects that a deployment running a version of Cloudera Manager earlier than 5.7

Operating System Support for Cloudera Manager

Altus Director 6.0 supports installation of Cloudera Manager and clusters on the following OS versions:

- For AWS and Google Cloud Platform only, Altus Director 6.0 now supports RHEL and CentOS version 7.5

Altus Director Release Notes

- For all cloud service providers, Altus Director 6.0 continues to support RHEL and CentOS versions 6.7, 6.8, 7.2, and 7.4

Encryption of application.properties File Content

You can now encrypt the values of the properties in the Altus Director application.properties file using a master password.

Additional Script Model

As of Altus Director 6.0, when you create bootstrap, post-create, or pre-terminate scripts, you can provide each script as an object consisting of an ID string and the script content.

For example:

```
bootstrapScripts: [  
  {  
    id: "SCRIPT-01"  
    content: ""#!/bin/bash ...  
  }  
]
```

You can use the new script model in addition to the previous model where you provide the script as an ordinary string.

Changes in Altus Director 6.0

Altus Director 6.0 has a number of changes that make it incompatible with the previous version. Review the following list of changes in Altus Director 6.0 before you upgrade to this version. The changes might require you to update your configuration files or modify the way that you use the product.

Support for New Azure VM Image

Altus Director 6.0 uses the *cloudera-centos-67-latest* VM image for Azure clusters.

If you have existing clusters in Azure that you need to grow or if you have an operation where you need to use an existing template that uses the *cloudera-centos-6-latest* image, you can add the image definition to the *images.conf* file so you can continue to use the *cloudera-centos-6-latest* image.

Add the following VM image definition to the *images.conf* file:

```
cloudera-centos-6-latest {  
  publisher: cloudera  
  offer: cloudera-centos-os  
  sku: 6_7  
  version: latest  
}
```

Restart Altus Director 6.0 to load the new image definition.

Custom *images.conf* files are stored in the Azure plug-in directory. If you create a custom *images.conf* file in Altus Director 6.0, add the file to the following directory:

```
/var/lib/cloudera-director-plugins/azure-provider-3.0.0/etc/
```

If you use other versions of the Azure plug-in for Director, update the *images.conf* file in the appropriate plug-in directory.

Storage Accounts in Azure Supported Only with CLI

As of Altus Director 6.0, all new instance templates for Azure use Managed Disks. On the Altus Director 6.0 web UI, you can no longer create instance templates that use storage accounts. To create a cluster in Azure that uses storage accounts, you must use the Altus Director CLI.

You can use the Altus Director CLI to create clusters with storage accounts in one of the following ways:

- Using an existing instance template that uses storage accounts.

Use an unedited instance template created in previous versions of Altus Director to create clusters with storage accounts. Editing an instance template that uses storage accounts can change the settings of the template. For more information, see [Edited or cloned Altus Director 2.8 instance templates for Azure do not work as expected](#) on page 20.

- Modifying the CLI configuration file to use storage accounts instead of managed disks in the instance template created in Altus Director 6.0.

You can modify the configuration file that you use with the CLI to use storage accounts instead of managed disks. In the *Instance Template* section of the configuration file, set the *managedDisks* field to `No`.

For more information about the *managedDisks* field, search for *managedDisks* in the [reference configuration file for Azure](#).

Stand-alone CLI Commands Removed

As of Altus Director 6.0, the stand-alone commands are removed from the Altus Director client. The Altus Director client is now used only for command line interactions with the Altus Director server. If an Altus Director server is not available, you cannot run the Altus Director CLI commands.

The following list shows the CLI commands that have been removed and the commands that you can use instead:

Obsolete Command	Replacement Command
bootstrap	bootstrap-remote
status	No replacement command. Use the server API or client libraries instead.
terminate	terminate-remote
update	No replacement command. Use the server API or client libraries instead.
validate	validate-remote

For more information about the Altus Director command line interface, see [Command Line Interface](#) on page 12.

Default H2 Database for Stand-alone CLI Not Created

Altus Director 6.0 does not create the H2 database used to maintain the local state of the stand-alone client. With the stand-alone CLI commands removed, Altus Director 6.0 no longer requires the H2 database.

Server API Versions v1 to v5 Not Available

Altus Director 6.0 introduces version d6.0 of the Altus Director server API. Along with server API d6.0, Altus Director 6.0 supports server API versions v6 through v12. It no longer supports Director server API version v5 or earlier.

If you use an API version v5 or earlier, you must update your applications to use API v6 or later.

Changes to Configuration Properties in *application.properties*

The following configuration properties in the *application.properties* file have been renamed in Altus Director 6.0:

Old property name	New property name
lp.terminate.deployment.clouderaManagerServerStopWaitTimeInMs	lp.cleanup.cm.stopWaitTimeInMs
lp.terminate.deployment.timeBetweenConnectionRetriesInMs	lp.cleanup.cm.timeBetweenConnectionRetriesInMs

If you customized the renamed properties in previous versions, you must rename the properties in the *application.properties* for Altus Director 6.0. Altus Director 6.0 ignores properties in *application.properties* file that use obsolete names.

Properties Removed from Cluster Configuration Files

When you use the CLI to launch a cluster, you can use a configuration file to customize the cluster setup. Altus Director provides sample configuration files that you can use for your own use.

When you use the CLI to launch a cluster, you can use a configuration file to customize the cluster setup. In Altus Director 6.0, a number of properties have been removed. If you use these properties in your cluster configuration file, update the file to use the correct property names. If you leave the property in the cluster configuration file, Altus Director ignores the property.

For more information about the cluster configuration file, see [The Altus Director Configuration File](#) on page 199.

As of Altus Director 6.0, the following properties in the cluster configuration file have been removed and replaced with another property:

Template	Removed Property	Replacement
Deployment	<i>customBannerText</i>	Use the <i>custom_banner_html</i> property of the CLOUDERA_MANAGER service.
Instance	<i>bootstrapScript</i>	<i>bootstrapScripts</i> Note the new property name is plural, ending in <i>s</i> .
Instance	<i>bootstrapScriptPath</i>	<i>bootstrapScriptsPaths</i> Note the new property name is plural, ending in <i>s</i> .

Kerberos Must be Configured Using Configuration Files

As of Altus Director 6.0, you cannot enable Kerberos for a new Cloudera Manager deployment on the Altus Director web UI. The option for enabling Kerberos for a new Cloudera Manager deployment on the web UI is removed.

You must use a configuration file to enable Kerberos.

Changes to the Altus Director SDK Library

As of Altus Director 6.0, the Java and Python client libraries released with Altus Director are generated using a new framework. The new framework generated changes in the client libraries. For example:

- In the Java client, enumerated values formerly represented by strings are now represented as Enum objects.
- In the Python client, API method names and model field names now use snake case (*get_status*) instead of camel case (*getStatus*).
- In the Python client, API exceptions are raised as APIException objects instead of HTTPError objects.

Code that uses Altus Director 2.x libraries does not work with the client libraries released with Altus Director 6.0. To use the new client libraries, you must update your code to match the changes in the client libraries.

You can continue to use Altus Director 2.x client libraries with Altus Director 6.0 through API versions v6 to v12. You need to update your code only if you want the code to work with the new client libraries released with Altus Director 6.0.

Changes to CDH 6.0 that affect Altus Director

CDH 6.0 has changes that affect Altus Director usage. The changes are not part of Altus Director, but they affect use of Altus Director.

External Spark 2 Parcels Not Supported in CDH 6.0

Prior to CDH 6.0, the use of Spark 2 requires the use of a separate parcel. This is no longer necessary in CDH 6.0, which includes Spark 2.2. However, use of a different version of Spark 2 is not supported with CDH 6.0 through the use of an external parcel. Existing Altus Director configuration files that bring in Spark 2 through a parcel must be modified for CDH 6.0 to rely on Spark 2.2 as packaged with CDH.

Sqoop 2 Not Supported in CDH 6.0

Sqoop 2 is not packaged with CDH 6.0. Bootstrapping CDH 6.0 clusters that call for Sqoop 2 will fail under Altus Director 6.0. Instead, use Sqoop 1 (service "SQOOP_CLIENT" and role "GATEWAY").

Sqoop 2 is still supported for CDH 5.x clusters, but deprecated.

Hue Service in CDH 6.0 Requires Python 2.7 and a Psycopg adapter

The Hue service included in CDH 6.0 requires a Python version that may not be available in the operating systems. The Hue service also requires a Psycopg adapter version that is different from the version used by the Cloudera Manager 6.0 agent package.

Altus Director provides a bootstrap script that you can use to install the correct version of Python and Psycopg adapter library for Hue. For more information about the bootstrap script, see [Hue service in CDH 6 that uses the wrong version of Python fails to start](#) on page 19.

Known Issues and Workarounds in Altus Director

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

Hue service in CDH 6 that uses the wrong version of Python fails to start

The Hue service included in CDH 6.x requires Python 2.7 and the [psycopg2](#) PostgreSQL database adapter library. The Psycopg adapter required by the Hue service is a newer version than the Psycopg adapter version used by the Cloudera Manager 6.0 agent package. The Python version required by Hue may be different from the versions provided in the operating systems.

If the Python and Psycopg adapter versions available to CDH 6.x and the Hue service are not the required versions, the Hue service fails to start. The error shows as a First Run failure where the Hue Server role fails to start, sending Altus Director into a failure state for the cluster.

Workaround: Altus Director provides [a bootstrap script](#) that installs the correct version of Python and the Psycopg adapter library. You can use the script to bootstrap CDH 6.x clusters with the Hue service.

You can use the bootstrap script in one of the following ways:

- Use the bootstrap script in a configuration file or in the Altus Director web UI.

For information about how to use the script file with a configuration file or the web UI, see the details about how the script works in the [readme file](#).

- Perform the work done in the bootstrap script on a preloaded AMI.

When the work is done ahead of time, the bootstrap script is not needed.

Cloudera Issue: DIR-8665

Incorrect migration of Cloudera Altus Director's database from H2 to MySQL causes upgrade to 2.8 to fail

The previously published documentation for migrating Cloudera Altus Director's data from H2 to MySQL resulted in an incorrect schema. This schema allowed Cloudera Altus Director to operate, but causes problems when upgrading Cloudera Altus Director. Re-migrate Cloudera Altus Director's data to a new MySQL database to fix the database schema and allow upgrading to Cloudera Altus Director 2.8.

Cloudera Issue: DIR-8507

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.

Cloudera Issue: DIR-5102

Edited or cloned Altus Director 2.8 instance templates for Azure do not work as expected

In Altus Director 6.0, the Azure instance template defaults to using managed disks.

If you edit and save an instance template created in Altus Director 2.8 that uses storage account, the saved template changes the settings to use managed disks instead of storage disks.

If you clone an instance template created in Altus Director 2.8 that uses storage account, the new template changes the settings to use managed disks instead of storage disks.

Workaround: Use instance templates created in Altus Director 2.8 without editing or cloning. If you need to change a template, use the CLI to deploy clusters with storage accounts.

Environment deletion fails with a 500 response code

Altus Director evicts terminated deployment, cluster, and external database server entities from its database before removing the environment. Altus Director also performs the same eviction on a periodic basis. A 500 response may be returned if the two evictions occur concurrently.

Workaround: Retry environment deletion.

Maximum length of environment name is misleading

Parsing of a Altus 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 Altus 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
```

Cloudera Issue: DIR-6301

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 Altus Director to AWS.

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

Cloudera Issue: DIR-4283

Altus Director cannot deploy Cloudera Navigator Key Trustee Server

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

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

Cloudera Issue: DIR-1757

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

If the Cloudera Manager username and password are changed directly in Cloudera Manager, Altus Director can no longer add new instances or authenticate with Cloudera Manager. Username and password changes must be implemented in Altus Director as well. For more information on keeping Altus Director and Cloudera Manager in sync, see [CDH Cluster Management Tasks](#).

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

Cloudera Issue: DIR-690

Altus Director might use AWS credentials from instance of Altus Director server

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

Severity: Medium

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

Cloudera Issue: DIR-1040

Root partition resize fails on CentOS 6.5 (HVM)

Altus 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](#).

Cloudera Issue: DIR-757

When using RDS and MySQL, Hive Metastore canary can fail in Cloudera Manager

If you include Hive in your clusters and configure the Hive metastore to be installed on MySQL, Cloudera Manager might 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 Altus 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 Altus Director from your platform's software repositories, select an older MySQL version that does not have this bug.

Cloudera Issue: DIR-923

Issues Fixed in Altus Director

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

Issues Fixed in Altus Director 6.0.1

LDAP authentication causes errors in usage-based billing

Altus Director can be configured to use a backing LDAP server, such as OpenLDAP or Active Directory, for its own user accounts. When configured as such, Altus Director fails to correctly collect usage information from itself that is needed to perform usage-based billing. This does not affect the normal functioning of Altus Director, but only causes the creation of malformed usage information that Cloudera is unable to process into billing records.

Cloudera Issue: DIR-8905

Incorrect Kerberos administrative password in rendered HOCON

In cases where Altus Director returns HOCON for a deployment template, such as exporting a cluster configuration file, the Kerberos administrative password, if present, was rendered instead as the Kerberos administrative username.

Cloudera Issue: DIR-8791

Issues Fixed in Altus Director 6.0.0

Simple setup with CentOS or RHEL 7.4 fails in some Azure regions

OS mappings for were not defined for azure-germany and azure-us-government regions. Attempts to bootstrap a cluster using a simple setup conf file that selected these combinations would result in a validation error. OS mappings have been added for Red Hat Enterprise Linux 7.4 and Cloudera-published CentOS 7.4 for Azure cloud environments where available.

Cloudera Issue: DIR-8671

UI Simple Setup ignores worker count

The UI Simple Setup form ignores the value in the worker count field and always uses the default value.

Cloudera Issue: DIR-8714

Custom Cloudera Manager service configurations not applied after cluster bootstrap or update

Cloudera Altus Director versions 2.7.x and 2.8.x fail to apply custom configuration properties for Cloudera Manager services, instead setting them back to automatic defaults as determined by Cloudera Manager. Common examples of affected configuration properties are memory settings for the Cloudera Manager service monitor and host monitor services.

Cloudera Issue: DIR-8470

Director fails to add worker nodes to a cluster that was created with Simple Setup

In previous versions, Director would fail to add worker nodes to a cluster that was created with Simple Setup. The error occurred because of the placement of ZooKeeper Server roles on worker nodes. The ZooKeeper Server role has been moved to the single master node.

Cloudera Issue: DIR-8699

Azure simple cluster setup fails when cluster name contains capital letters

Azure simple setup cluster installation fails when the cluster name contains capital letters.

Cloudera Issue: DIR-8335

Terminating a deployment fails when the Cloudera Manager instance has been deleted without using Altus Director

If the cloud instance hosting Cloudera Manager for a deployment is deleted without going through Altus Director, then deployment termination might fail while attempting to clean up the Kerberos credentials associated with Cloudera Manager. This failure only occurs when Altus Director cannot contact Cloudera Manager directly over its web port and must instead use an SSH tunnel to reach it.

Cloudera Issue: DIR-8331

Autorepair on Azure can remove healthy instances

An Azure cluster with autorepair enabled can have healthy instances removed if there's a failure to communicate with Azure.

Cloudera Issue: DIR-8291

Erroneous error code 406 and error message returned when simple setup cluster is launched with incorrect data

When attempting to launch a cluster with the simple setup procedure, Altus Director server might return an HTTP error code of 406 with the message: **Could not find acceptable representation**. This occurs when there is an error in the data supplied for the cluster, and the server is unable to formulate a correct error response. This error code and message should not occur when the data for simple cluster setup is correct.

Cloudera Issue: DIR-8241

FirstRun command occasionally fails

When executing FirstRun, Director's list of commands was correct, but misordered. Due to the timing, this results in infrequent failures during FirstRun while bootstrapping clusters.

Cloudera Issue: DIR-7977

Swagger model schema for external databases is incorrect

The Swagger model schema for external databases is not displayed correctly.

Cloudera Issue: DIR-941

Unsupported Components and Features in Altus Director

This page lists the unsupported features in Altus Director.

- Key Trustee KMS
- Cloudera Navigator Key Trustee Server (KTS)
- Migration of highly available MapReduce JobTracker roles
 - MapReduce JobTracker is a MR1 role. Cloudera recommends use of YARN/MR2 to run MapReduce jobs instead of MR1.

Requirements and Supported Versions

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

Cloud Providers

Altus Director has native support for Amazon Web Services (AWS), Google Cloud Platform, and Microsoft Azure.

Each Altus Director release embeds the current plug-in for supported cloud providers, but a newer plug-in might have been posted on the Cloudera GitHub site subsequent to the Altus 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](#)

Altus Director Service Provider Interface (SPI)

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

Supported Software and Distributions

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

	Altus Director	Cloudera Manager and CDH
Operating Systems (64-bit only)	RHEL and CentOS 6.7, 6.8, 7.2, 7.4, 7.5 Ubuntu 14.04, 16.04	<p>For AWS: RHEL and CentOS 6.8, 6.9, 7.2, 7.3, 7.4</p> <p>For Google Cloud Platform: RHEL and CentOS 6.8, 6.9, 7.2, 7.3, 7.4</p> <p>For Microsoft Azure: RHEL and CentOS 6.8, 6.9, 7.2, 7.3, 7.4</p> <p>Notes:</p> <ul style="list-style-type: none"> • RHEL 7.2 is supported only for Cloudera Manager and CDH 5.7 and higher. • RHEL 7.3 is supported only for Cloudera Manager and CDH 5.10 and higher, and only on AWS and Google Cloud Platform. • RHEL 7.4 is supported only for Cloudera Manager and CDH 5.11 and higher. • AWS supports Spot instances only on Centos operating systems, not on RHEL.

	Altus Director	Cloudera Manager and CDH
		<ul style="list-style-type: none"> 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 this instance type.
Oracle Java SE Development Kit (JDK)	<p>Oracle JDK version 8</p> <p>For download and installation information, see Java SE Downloads.</p>	<p>Oracle JDK version:</p> <ul style="list-style-type: none"> Cloudera Manager and CDH 5.x: JDK 7 or 8. Cloudera Manager and CDH 6: JDK 8. <p>For more information about deploying Java in Altus Director-managed clusters, see Deploying Java on Cluster Instances on page 169.</p>
Default Database	<p>Embedded H2 database (not recommended for production use)</p> <p>Recommended size: 10 GB. For information on shrinking the size of the database, see Shrinking an H2 Database on page 122.</p> <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;"> <p> Note: Less storage space might suffice. The required size of the Altus Director database depends on how many environments, deployments, and clusters Altus Director manages.</p> </div>	<p>Embedded PostgreSQL Database (not recommended for production use)</p>
Supported Databases	<p>MySQL 5.5, 5.6, 5.7</p> <p>MariaDB 5.5</p> <p>Recommended size: 10 GB.</p> <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;"> <p> Note: Less storage space might suffice. The required size of the Altus Director database depends on how many environments, deployments, and clusters Altus Director manages.</p> </div>	<p>MySQL 5.5, 5.6, 5.7</p> <p>MariaDB 5.5</p> <p>PostgreSQL 8.1, 8.3, 8.4, 9.1, 9.2, 9.3, 9.4, 9.5, 9.6</p>



Note: In production environments, you should use an external MySQL or MariaDB database for Altus Director. For information on using an external MySQL database in place of the H2 embedded database, see [Using MySQL for Altus Director Server](#) on page 102. For information on using an external MariaDB database in place of the H2 embedded database, see [Using MariaDB for Altus Director Server](#). By default, Altus 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 Altus 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 6 and CDH 6. 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 124.



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 Altus Director.

	Altus Director	Cloudera Manager and CDH
CPU	2	4
RAM	3.75 GB	64 GB
Disk	8 GB	500 GB
Recommended AWS instance	c3.large or c4.large	Cloudera Manager: m4.xlarge or m4.4xlarge
Recommended Google Cloud Platform instance	n1-standard-2	n1-highmem-4 or n1-highmem-8
Recommended Microsoft Azure instance	Standard_D3 or larger	The following Azure instance types are supported: <ul style="list-style-type: none"> Standard_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

	Altus Director	Cloudera Manager and CDH
		<ul style="list-style-type: none"> 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 might not be available in every region. Altus Director does not dynamically validate instance type by region. Contact your Cloudera account representative for more information.

Supported Cloudera Manager and CDH Versions

- Altus Director 6 can install any version of Cloudera Manager 6 with any CDH 6 parcels. Use of CDH packages is not supported.
- Altus Director 6 can install Cloudera Manager 5.7 (and higher) and the [corresponding supported versions of CDH](#). Use of CDH packages is not supported.
- Altus Director 2.x cannot be used to install Cloudera Manager 6 and CDH 6.

If you are using Altus Director 6 to deploy Cloudera Manager and CDH, the latest released version of Cloudera Manager 6 and CDH 6 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 [Advanced Setup: Installing Cloudera Manager and CDH on AWS](#) on page 43.
- For Google Cloud Platform, see [Deploying Cloudera Manager and CDH on Google Compute Engine](#) on page 56.
- For Microsoft Azure, see [Deploying Cloudera Manager and CDH on Microsoft Azure](#).

Cloudera Data Science Workbench Support

Altus Director can deploy Cloudera Data Science Workbench on clusters running Cloudera Manager 5.13.1 (and higher 5.x versions).

Cloudera Data Science Workbench (1.4.x and lower) is not currently supported with Cloudera Manager 6.0.0 and CDH 6.0.0. Consequently, you cannot use Altus Director 6 to deploy Cloudera Data Science Workbench on a C6 cluster. Cloudera Data Science Workbench will be supported with Cloudera Enterprise 6 in a future release.

Networking and Security Requirements

Altus Director recommends the following inbound ports to be open:

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

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



Note: 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 Altus Director instance. This is done in a later step.

In a restricted network environment, you might 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 Altus Director](#).

Supported Browsers

Altus Director supports the following browsers:

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

Getting Started with Altus Director

This section explains how to get Altus Director up and running on Amazon Web Services (AWS), Google Cloud Platform, and Microsoft Azure.

Getting Started on Amazon Web Services (AWS)

To use Altus Director on AWS, you create an environment in Amazon Virtual Private Cloud (Amazon VPC), start an instance in AWS to run Altus Director, and create a secure connection. This section describes the steps for each of these tasks.



Important:

Altus 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 Altus Director, see [Using Spot Instances](#).

Preparing Your AWS EC2 Resources

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

Setting Up a VPC

Altus 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.

Perform the following steps:

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

Altus Director requires the following inbound ports to be open:

Table 1: Ports

Type	Protocol	Port Range	Source
ALL Traffic	ALL	ALL	<i>security_group_id</i>
SSH (22)	TCP (6)	22	<your ip address>



Note: By default, Altus Director requires unrestricted outbound connectivity. You can configure Altus Director to use proxy servers or a local mirror of all the relevant repositories if required.

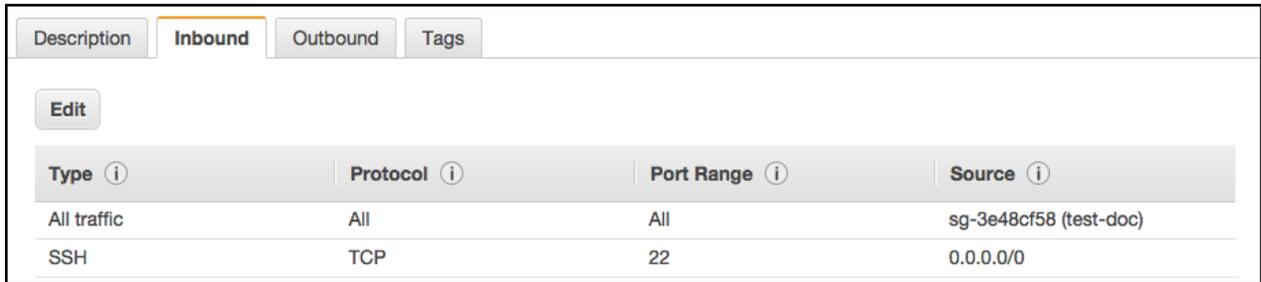
Creating a New Security Group

The simplest way to set up the required network connectivity for Altus 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 [Ports](#) table.

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



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

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

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



Note: Alternatively, you can install Altus Director in a different region, on a different cloud provider, or a different network environment. For information on these more complex setups, see [Running Altus Director and Cloudera Manager in Different Regions or Clouds](#) on page 99.

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 Altus 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 24.
 - 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.

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

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

The screenshot shows the AWS Management Console interface for selecting an AMI. On the left, there is a sidebar with 'Community AMIs' selected under 'Operating system'. The search box contains 'rhel-7.3 HVM'. The results show three RHEL 7.3 HVM AMIs from Red Hat, Inc. with 'Select' buttons.

AMI ID	Provider	Root device type	Virtualization type	Architecture
RHEL-7.3_HVM_GA-20161026-x86_64-1-Hourly2-GP2 - ami-b63769a1	Provided by Red Hat, Inc.	ebs	hvm	64-bit
RHEL-7.3_HVM_Beta-20160819-x86_64-2-Hourly2-GP2 - ami-96fe9d81	Provided by Red Hat, Inc.	ebs	hvm	64-bit
RHEL-7.3_HVM_RC1-20161005-x86_64-1-Hourly2-GP2 - ami-d8470bcf	Provided by Red Hat, Inc.	ebs	hvm	64-bit

- c. Click **Select** for the AMI version you choose.



Note: For more information on finding AMIs, see [Choosing an AMI](#).

4. Select the instance type for Altus 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**. Altus 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).

Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. A copy of a tag can be applied to volumes, instances or both. Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key (127 characters maximum)

Value (255 characters maximum)

Instances
i

Volumes
i

This resource currently has no tags

Choose the Add tag button or [click to add a Name tag](#).
Make sure your [IAM policy](#) includes permissions to create tags.

Add Tag (Up to 50 tags maximum)

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 Altus 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 Altus 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 [Preparing Your AWS EC2 Resources](#) on page 29.
- b. Click the check box to acknowledge that you have access to the private key.

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

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

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

Select a key pair

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

Cancel
Launch Instances

12 Click **Launch Instances**.

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

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

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: Altus Director, CDH, and Cloudera Manager support only 64-bit Linux. For CDH and Cloudera Manager on Amazon EC2, Altus 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: Altus 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.

Installing Altus Director Server and Client on the EC2 Instance

To install Altus 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 Altus 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` (RHEL) or `centos` (CentOS) into the EC2 instance you created for Altus 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 might 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 Altus Director host. Altus Director 6 supports JDK version 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
```

- 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
```

- Add the Altus Director repository to the package manager:

```
cd /etc/yum.repos.d/
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat7/cloudera-director.repo"
```

- Install Altus Director server and client by running the following command:

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

- Start the Altus Director server by running the following command:

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

- If the RHEL 7 or CentOS firewall is running on the EC2 instance where you have installed Altus 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

- SSH as `ec2-user` into the EC2 instance you created for Altus 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
```

- Install a supported version of the Oracle Java Development Kit (JDK) on the Altus Director host. Altus Director 6 supports JDK version 8. For download and installation information, see [Java SE Downloads](#). After downloading the RPM file to the EC2 instance, install the JDK:

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

- Add the Altus Director repository to the package manager:

```
cd /etc/yum.repos.d/
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat6/cloudera-director.repo"
```

- Install Altus Director server and client by running the following command:

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

- Start the Altus Director server by running the following command:

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

- 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 Altus 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 Altus Director repository to the package manager and the signing key.

If you are on version 14.04, run the following commands:

```
cd /etc/apt/sources.list.d/  
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1404/cloudera-director.list"  
-O  
sudo curl -s -L "https://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1404/archive.key" | sudo  
apt-key add -
```

If you are on version 16.04, run the following commands:

```
cd /etc/apt/sources.list.d/  
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1604/cloudera-director.list"  
-O  
sudo curl -s -L "https://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1604/archive.key" | sudo  
apt-key add -
```

3. Install a supported version of the Oracle Java Development Kit (JDK) on the Altus Director host. Altus Director 6 supports JDK version 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-j2sdk1.8
```

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

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

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

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

6. 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 Altus Director Server or Altus 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 Altus Director. Optionally, you can install just the client, and configure it to interact with an Altus Director server running on another machine, rather than the default `localhost`. 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 Altus 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 Altus 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 Altus 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 Altus 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 Altus 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 Altus 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 Altus 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 Altus 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 Altus Director-accessible host using the private IP address or internal fully qualified domain name (FQDN). For example, when you connect to the Altus 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 Altus 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';  
}
```



Note: If you copy the function above, be sure to replace 172.31 with the proper prefix value for your VPC subnet.

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](#).

Adding a Altus Director Environment on AWS

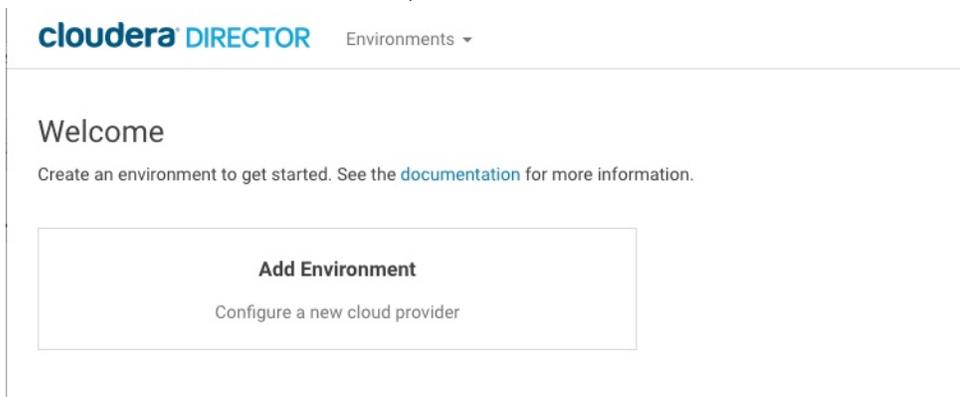
To deploy Cloudera Manager and CDH on an AWS EC2 instance, you begin by adding an environment in Altus Director. The environment defines common settings, like region and key pair, that Altus Director uses with AWS. While adding an environment, you are also prompted to deploy its first cluster.

To add your first environment in a Altus Director instance:

1. Open a web browser and go to the private IP address of the instance you created in [Launching an EC2 Instance for Altus Director](#) on page 30. Include port 7189 in the address. For example:

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

2. In the **Altus Director** login screen, enter `admin` in both the **Username** and the **Password** fields.
3. In the Altus Director **Welcome** screen, click the **Add Environment** tile.



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 Altus Director instance was created.

cloudera DIRECTOR Environments ▾

Add Environment

General Information

Environment name * ?

Cloud provider **Amazon Web Services (AWS)** ?

Access key ID ?

Secret access key ?

EC2 (Elastic Compute Cloud)

EC2 region ?

> Advanced Options

RDS (Relational Database Service)

RDS region ?

> Advanced Options

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

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

 **Note:** Depending on the operating system, different AMIs might 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. This example indicates that `centos` is the username to be used with this Centos AMI:



- b. Copy your SSH private key (for example, the key you created in [Launching an EC2 Instance for Altus Director](#) on page 30) in the **Private key** field.

SSH credentials

Username * ?

Private key * File Upload Direct Input ?

- 5. Click **Continue** to add the environment and advance to creating either a Simple Setup cluster or an Advanced Setup cluster:

Environment successfully created. ✕

Simple Setup

Use default settings to create a cluster.

Advanced Setup

Define advanced settings and topology.

Creating a cluster with the **simple setup** procedure is quick and easy because many configuration choices have been made for you. This is a great way to try out the product, and get a cluster up and running quickly. Creating a cluster with the **advanced setup** procedure gives you many more options for how you want your cluster to be configured, and can be used to set up advanced features, like Kerberos, TLS, and external databases.

To proceed, and create one of these kinds of clusters, choose one of the following topics:

- [Simple Setup: Creating a Cluster on AWS with Default Settings](#) on page 41
- [Advanced Setup: Installing Cloudera Manager and CDH on AWS](#) on page 43

Simple Setup: Creating a Cluster on AWS with Default Settings

The simple setup procedure provides the simplest and most reliable way to get a Cloudera Manager deployment and CDH cluster up and running quickly when you do not require custom configurations or advanced features, like Kerberos, TLS, or external databases. To bootstrap a new cluster, you need only provide some information about your cloud provider of choice, and about the type of cluster you want to create. All of the other details about how your cluster is configured, like its topology and versions of Cloudera Manager and CDH, are determined for you by Altus Director.

Simple setup clusters are not recommended as production clusters. But a good strategy for creating a production cluster is to export a client configuration file for a simple setup cluster, and then edit that configuration file to add more advanced features. In this way, you simplify the task of creating a custom cluster, beginning from a known working configuration, rather than starting from the more complete, but more complex [aws.reference.conf](#) file. You can export a client configuration file through the web UI or through the server API. For more information, see [Exporting a Configuration File](#) on page 201.

Type of Simple Setup Clusters

Choose from five different types of cluster with Simple Setup, based on the workloads you will run in the cluster. The workload type you choose determines the services in the cluster.

- **Basic:** Provides a simple Hadoop environment. Includes Hive and MapReduce.
- **Data Engineering:** For Spark workloads and ETL jobs. Includes Hive and Spark.
- **Analytic Database:** For business intelligence and SQL analytics. Includes Impala and Hive.
- **Operational Database:** For NoSQL real-time application serving. Includes HBase.
- **Enterprise Data Hub:** Provides a comprehensive range of services for the Cloudera platform.

Ways to Create a Simple Setup Cluster

There are several ways to create a cluster with the simple setup procedure:

- Use the Altus Director UI, and click the tile **Simple Setup** at the conclusion of the **Add Environment** procedure, or navigate to an existing environment and clicking the **Add Cluster** button.
- Use the Altus Director CLI and the `bootstrap-remote` command with a configuration file. You can define a simple client configuration file with the same environment and cluster information that you would supply through the web user interface. For a sample configuration file, including instructions for creating a cluster, see [aws.simple_setup.conf](#) on the Cloudera GitHub site.
- Submit JSON or HOCON input to the Altus Director API import endpoint. For details, see `importClientConfig` in the Altus Director API console at `director_ip_address:port/api-console/index.html`.

Launching a Simple Setup Cluster with the Altus Director UI

To launch a cluster using the simple setup procedure with the Altus Director UI, perform the following steps:

1. Add a Altus Director environment, as described in [Adding a Altus Director Environment on AWS](#) on page 39.
2. Begin the simple setup cluster installation in one of the following ways:
 - Click the **Simple Setup** tile at the conclusion of the **Add Environment** procedure.
 - Navigate to an existing environment and click the **Add Cluster** button.
3. Provide values for the fields on the **Add Cluster** screen:
 - a. **Cluster name:** Between 2 and 40 alphanumeric characters. Space, underscore, and hyphen are allowed except at the beginning or end of the name. EC2 instances for the cluster nodes are prefixed with the cluster name.
 - b. **Workload Type:** Select the typical workload that this cluster will run on from the drop-down list. The workload type is used to determine the services that run inside the cluster.
 - c. **Worker Node Instance Type:** m4.xlarge or larger is recommended.
 - d. **Worker Node Count:** Number of cluster nodes for worker roles.
 - e. **Security group IDs:** Specify a list of one or more security group IDs. Must begin with `sg-`.
 - f. **VPC subnet ID:** The AWS virtual private cloud (VPC) subnet ID.
4. Click **Continue** to launch the cluster.

If you launch additional clusters with the simple setup procedure using the same Altus Director environment, the fields on the **Add Cluster** screen for Worker Node Instance Type, Worker Node Count, Security Group IDs, and VPC subnet ID will be pre-populated with the values you used for your previous simple setup cluster.

Converting a Configuration File from Simple to Standard Format

When you launch a cluster using the Altus Director CLI and the `bootstrap-remote` command with a simple configuration file, your configuration file is automatically converted from a simple cluster configuration file to a standard one. If you want to see your configuration file converted from simple to standard format before you have used it, run the CLI command `convert-remote`.



Note: For security reasons, some fields in the converted configuration file will be redacted. Values for the redacted fields must be provided in order to use the configuration file:

- Cloud provider credentials
- SSH private key
- Any database passwords

Replace the sections `REDACTED` and `---BEGIN RSA PRIVATE KEY---` with actual values to make the converted configuration file usable.

The default name for files created with the `convert-remote` command is `director.out`. Use the `lp.cli.file.outputFilename` property if you want to give the output file a different name. You can do this in the CLI `application.properties` file (`/etc/cloudera-director-client/application.properties` on the

Altus Director instance). Another way is to pass your desired file name on the command line with the `convert-remote` command:

```
cloudera-director convert-remote simple.conf \
  --lp.cli.file.outputFilename=myoutputfilename.conf \
  --lp.remote.username=admin --lp.remote.password=admin
```

This method enables you to provide a different file name each time you convert a configuration file from simple to standard format.

Advanced Setup: Installing Cloudera Manager and CDH on AWS

This section describes how to create a cluster using the advanced setup procedure in the Altus Director UI. For the simple setup procedure, see [Simple Setup: Creating a Cluster on AWS with Default Settings](#) on page 41.



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, Altus Director cannot grow the cluster, for example, 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 192.



Note: In most AWS regions, Altus Director assigns a tag during the creation of each instance it creates to facilitate instance management. The GovCloud (US) and China (Beijing) regions do not support tagging of instances on creation, so for instances in these regions, the tag is created after the instance is created. If you are running Altus Director in the GovCloud (US) or China (Beijing) regions, you must turn off `useTagOnCreate` in the Altus Director AWS plugin. See [Configuring Tag-on-create for AWS GovCloud \(US\) and China \(Beijing\) Regions](#) in the Troubleshooting section for detailed instructions.

To install a Cloudera Manager deployment and launch a CDH cluster:

1. Add an Altus Director environment, as described in [Adding a Altus Director Environment on AWS](#) on page 39.
2. Open a web browser and go to the private IP address of the instance you created in [Launching an EC2 Instance for Altus Director](#) on page 30. Include port 7189 in the address. For example:

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

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

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

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 Altus Director](#) on page 30, 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 192.

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



Note: Altus Director uses the **Name** field on AWS. By default, Altus Director puts the following values in the **Name** field: Altus Director id, Altus Director template name. If you want to create a custom key value for this field in place of "Name," see [Configuring Altus 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 30.

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 Altus 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. Note that AWS requires user data to be base64-encoded. You must perform the base64-encoding manually. For more information, including examples, see [Instance Metadata and User Data](#) in the AWS documentation.

h. Click **Save changes**.

Instance Template
✕

Instance Template name

Instance type ?

Image (AMI) ID ?

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

Security group IDs - + ?

VPC subnet ID ?

➤ **Advanced Options**

Cancel Save changes

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

- Cloudera Enterprise: includes the core CDH services (HDFS, Hive, Hue, MapReduce, Oozie, Sqoop 1, YARN, and ZooKeeper) and, depending on the license edition, one or more additional services (Accumulo, HBase, Impala, Navigator, Solr, Spark). For more information on Cloudera Enterprise licenses, see [Managing Licenses](#) in the Cloudera Manager documentation.

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

Licensing

Desired License Type * ?

Please provide a Cloudera Manager license key.

License Key * File Upload Direct Input ?

Billing ID ?

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

1. In the **Desired License Type** field, select **Cloudera Enterprise**.
 2. In the **License Key** field, either select a Cloudera Enterprise license file to upload or select **Direct Input** and input the license file text directly into the text area.
 3. To enable usage-based billing, enter the billing ID provided to you by Cloudera in the **Billing ID** field.
9. 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**:

Database Server

?

Configurations (optional)

?

?

?

Cloudera Manager Admin Username ?

Cloudera Manager Admin Password ?

Re-enter Password ?

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

- [Using MySQL for Altus Director Server](#) on page 102
- [Using MariaDB for Altus Director Server](#) on page 115

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**.

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

Configuration	Value	Scope
firehose_heapsize	1073741824	Host Monitor
firehose_heapsize	1073741824	Service Monitor

At the bottom of the dialog, there are buttons for 'Cancel', 'Reset', and 'Save Changes'.

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

Altus Director version	Cloudera Manager version installed
Altus Director 2.0	Latest released version of Cloudera Manager 5.5
Altus Director 2.1	Latest released version of Cloudera Manager 5.7
Altus Director 2.2	Latest released version of Cloudera Manager 5.8
Altus Director 2.3	Latest released version of Cloudera Manager 5.10
Altus Director 2.4	Latest released version of Cloudera Manager 5.11
Altus Director 2.5	Latest released version of Cloudera Manager 5.12
Altus Director 2.6	Latest released version of Cloudera Manager 5.13
Altus Director 2.7	Latest released version of Cloudera Manager 5.14
Altus Director 2.8	Latest released version of Cloudera Manager 5.15
Altus Director 6.0	Latest released version of Cloudera Manager 6.0

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 <https://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 https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.5.4/.



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

- c. In the **Repository Key URL** field, enter the URL for the repository key. Repository key URLs have the same form as repository URLs except they end with the name of the key file instead of the Cloudera Manager version. For example, the repository key URL for any version of Cloudera Manager 5 on any supported version of RHEL 7 is https://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 Altus Director you are using:

Altus Director version	CDH version installed
Altus Director 2.0	Latest released version of CDH 5.5
Altus Director 2.1	Latest released version of CDH 5.7
Altus Director 2.2	Latest released version of CDH 5.9
Altus Director 2.3	Latest released version of CDH 5.10
Altus Director 2.4	Latest released version of CDH 5.11
Altus Director 2.5	Latest released version of CDH 5.12
Altus Director 2.6	Latest released version of CDH 5.13
Altus Director 2.7	Latest released version of CDH 5.14
Altus Director 2.8	Latest released version of CDH 5.15
Altus Director 6.0	Latest released version of CDH 6.0

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 <https://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 <https://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 192.

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

- In the **Create New Instance Template** modal screen, click **Advanced Options**.
- In the **Spot bid (USD/hr)** field, enter your Spot bid price.
- Click the **Use Spot instances** checkbox.
- Click **Save Changes**.

For more information about using Spot instances with Altus Director, see [Using Spot Instances](#) on page 202.

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

16 Click **Continue**.

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

Status

TESTCLUSTER01 Bootstrapping

7 / 30

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

- Starting
- Starting
- Starting

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

You are finished with the deployment tasks.

Pausing a Cluster on 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 Altus Director, terminate your instances to stop incurring charges to your AWS account.

1. In Altus 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 Altus Director (the configuration file or database, for example), back it up.

3. In the AWS Management Console, terminate the Altus Director instance and any other instance Altus Director was unable to terminate.
4. If applicable, terminate any external database you configured Altus Director to use.

Getting Started on Google Cloud Platform

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



Important: Altus 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 Altus 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 Altus Director, Cloudera recommends that you configure some tools for your Google Cloud Platform account.

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

Creating a Service Account for Altus Director

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

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

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



Google Compute Engine

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

[Learn more](#)

[Try this API in APIs Explorer](#)

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



Create service account

Key type

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

JSON
Recommended

P12
For backward compatibility with code using the P12 format



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

Creating and Uploading an SSH Key

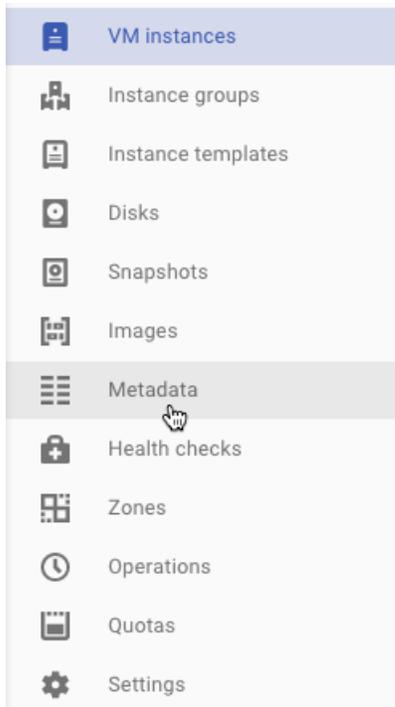
To SSH into an instance using your own terminal (as opposed to the Google Cloud Platform console), you must generate and upload an SSH key.

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 2: VM Instance Values

Name	Description	Details/Restrictions
Name	Name of the instance.	The name must start with a lowercase letter followed by up to 62 lowercase letters, numbers, or hyphens. The name cannot end with a hyphen.
Zone	Where your data is stored.	Some resources can only be used by other resources in the same zone or region. For example, to attach a disk to a VM instance, both resources must reside in the same

Name	Description	Details/Restrictions
		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 Altus 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 Altus Director. For more information about supported Linux versions, see Supported Software and Distributions on page 24.
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 Altus Director Server and Client](#) on your instance.

Installing Altus Director Server and Client on Google Compute Engine

Cloudera recommends that you install Altus Director server on your cloud provider in the subnet where you will create CDH clusters, because Altus Director must have access to the private IP addresses of the instances that it creates. To install Altus 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 Altus Director host. Altus Director 6 supports JDK version 8.



Note: Spark 2.2 and higher requires Python 2.7 or higher.

For Java installation information, see [Java SE Downloads](#).

3. Download Altus Director by running the following commands:

```
cd /etc/yum.repos.d/  
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat6/cloudera-director.repo"
```

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

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

5. Start the Altus 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 Altus Director host. Altus Director 6 supports JDK version 8. For installation information, see [Java SE Downloads](#).
3. Download Altus Director by running the following commands:

```
cd /etc/yum.repos.d/
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat7/cloudera-director.repo"
```

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

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

5. Start the Altus 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 Altus Director host. Altus Director 6 supports JDK version 8. For installation information, see [Java SE Downloads](#).
3. Download Altus Director and add the signing key.

If you are on version 14.04, run the following commands:

```
cd /etc/apt/sources.list.d/
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1404/cloudera-director.list"
-o
sudo curl -s -L "https://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1404/archive.key" | sudo
apt-key add -
```

If you are on version 16.04, run the following commands:

```
cd /etc/apt/sources.list.d/
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1604/cloudera-director.list"
-o
sudo curl -s -L "https://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1604/archive.key" | sudo
apt-key add -
```

Getting Started with Altus Director

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

```
apt-get update
apt-get install cloudera-director-server
apt-get install oracle-j2sdk1.8
```

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

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

6. 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 Altus 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 Altus 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 52. Include port 7189 in the address. For example:

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

2. In the **Altus Director** login screen, enter `admin` in both the **Username** and the **Password** fields.
3. In the Altus 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 50.
- 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 50.

Add Environment

GENERAL INFORMATION

Environment name * ?

Cloud provider ?

Project ID * ?

▼ **Advanced Options**

Client ID JSON Key File Upload Direct Input ?

```
e/iN4x1KlAinCDXskL+tiVn6uchSs
0M57r/p5
u89wUt5zk7
/vX1xNhp9sQiuTzs6KtYTSnrK9GwdQ2fqBAoG
Af0mVgn4WgKZ6TamDriFBL4ocAaBx
ih36YNgHy736Ax13AHQ19Mna14QA
/2dSQ0CQ031
/MdssSufqhSeMA1ht0IZpdz3xNrBsms84G4Y1
4QgAqiKV0QMUYkBB9tgnLdL75m58xDHEe0UM
yyqGm+AryQPW35B1Ak4CMWEeWTQYDo=
```

e. In the **Advanced Options** section, enter the same **region** that your Altus 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 51 in the **Private key** field.

GOOGLE COMPUTE ENGINE

▼ **Advanced Options**

Region ?

SSH CREDENTIALS

Username * ?

Private key * File Upload Direct Input ?

```
cr4gwiNsk/rQMx06J+I9h0ij2ToHd
sqGZJ/MmhD6vjfInbpbNw54121N8K08Pe63
Hkx4lsUlqD/02ZyieA/vTdVsvm+v7+tpw
WD5Df4QohYRmlf2kRdIuG5XTjoxscykfPTO
6Dq9DH6JkJSSX6BaAl
yFu/ZebgEp20psMqANv6sJgkT5ic
Lcn+6C2TbsdTLdkpZ5veONGe ih9h0S
0i3SL7fjyy0+w6YnqTMAvqh1BscWB
TbqnfZzEKkxHikaBuUeTI
eYrQFTEtg8XkpgyTQRNe01DaHycUN
```

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

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

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

The **Instance Template** modal screen displays.

Add Cloudera Manager

Environment TESTENV01

Cloudera Manager name ?

Instance Template ?

Select a Template

Create New Instance Template

Database Server ?

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

- In the **Instance Template name** field, enter a name for the template.
- In the **Instance type** field, select **n1-highmem-4** or **n1-highmem-8**.
- In the **Machine type** field, enter the machine type you chose in [Creating a Google Compute Engine VM Instance](#) on page 52.

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

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

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

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

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

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

Configuration	Value	Scope
firehose_heapsize	1073741824	Host Monitor
firehose_heapsize	1073741824	Service Monitor

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

Altus Director version	Cloudera Manager version installed
Altus Director 2.0	Latest released version of Cloudera Manager 5.5
Altus Director 2.1	Latest released version of Cloudera Manager 5.7
Altus Director 2.2	Latest released version of Cloudera Manager 5.8
Altus Director 2.3	Latest released version of Cloudera Manager 5.10
Altus Director 2.4	Latest released version of Cloudera Manager 5.11
Altus Director 2.5	Latest released version of Cloudera Manager 5.12
Altus Director 2.6	Latest released version of Cloudera Manager 5.13
Altus Director 2.7	Latest released version of Cloudera Manager 5.14
Altus Director 2.8	Latest released version of Cloudera Manager 5.15
Altus Director 6.0	Latest released version of Cloudera Manager 6.0

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 <https://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 https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.5.4/.



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

- c. In the **Repository Key URL** field, enter the URL for the repository key. Repository key URLs have the same form as repository URLs except they end with the name of the key file instead of the Cloudera Manager version. For example, the repository key URL for any version of Cloudera Manager 5 on any supported version of Red Hat 7 is https://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 Altus Director you are using:

Altus Director version	CDH version installed
Altus Director 2.0	Latest released version of CDH 5.5
Altus Director 2.1	Latest released version of CDH 5.7
Altus Director 2.2	Latest released version of CDH 5.9
Altus Director 2.3	Latest released version of CDH 5.10
Altus Director 2.4	Latest released version of CDH 5.11
Altus Director 2.5	Latest released version of CDH 5.12
Altus Director 2.6	Latest released version of CDH 5.13
Altus Director 2.7	Latest released version of CDH 5.14
Altus Director 2.8	Latest released version of CDH 5.15
Altus Director 6.0	Latest released version of CDH 6.0

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 <https://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 <https://archive.cloudera.com/cdh5/parcels/5.4.8>.



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

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

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

14 Click **Continue**.

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

Status

TESTCLUSTER01 Bootstrapping

7 / 30

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

1. Starting
2. Starting
3. Starting

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

You are finished with the deployment tasks.

Cleaning Up Your Google Cloud Platform Deployment

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

1. In Altus 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 Altus Director (the configuration file or database, for example), back it up.
3. In the Google Compute Console, delete the Altus Director instance and any other instance Altus Director was unable to delete.
4. If applicable, delete any external database you configured Altus Director to use.

Getting Started on Microsoft Azure

Before you can use Altus 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 Altus Director

To get started with Altus Director and Microsoft 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 Altus Director uses the service principal credentials to create and delete resources on Microsoft Azure.

Getting Started with Altus Director

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 Altus 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

Altus 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 might 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.



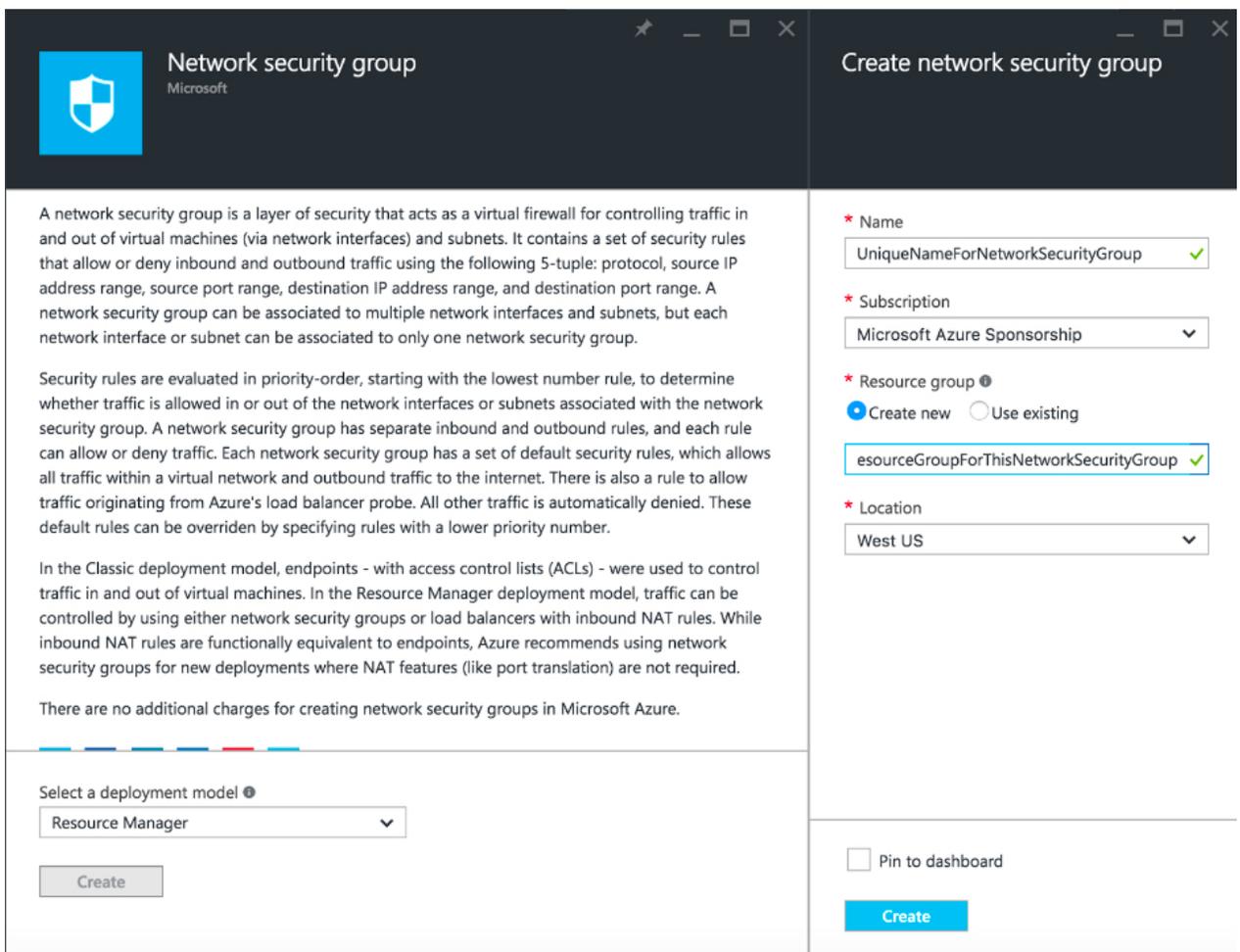
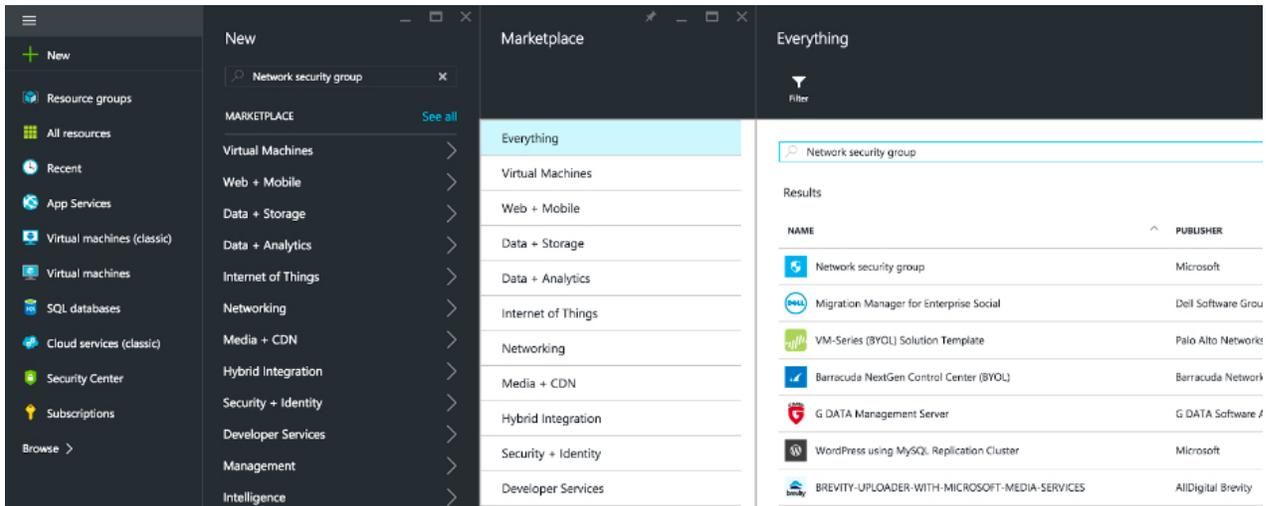
Note: You must open the following ports if you are allowing access to the web UI from public IP addresses:

- Altus 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.



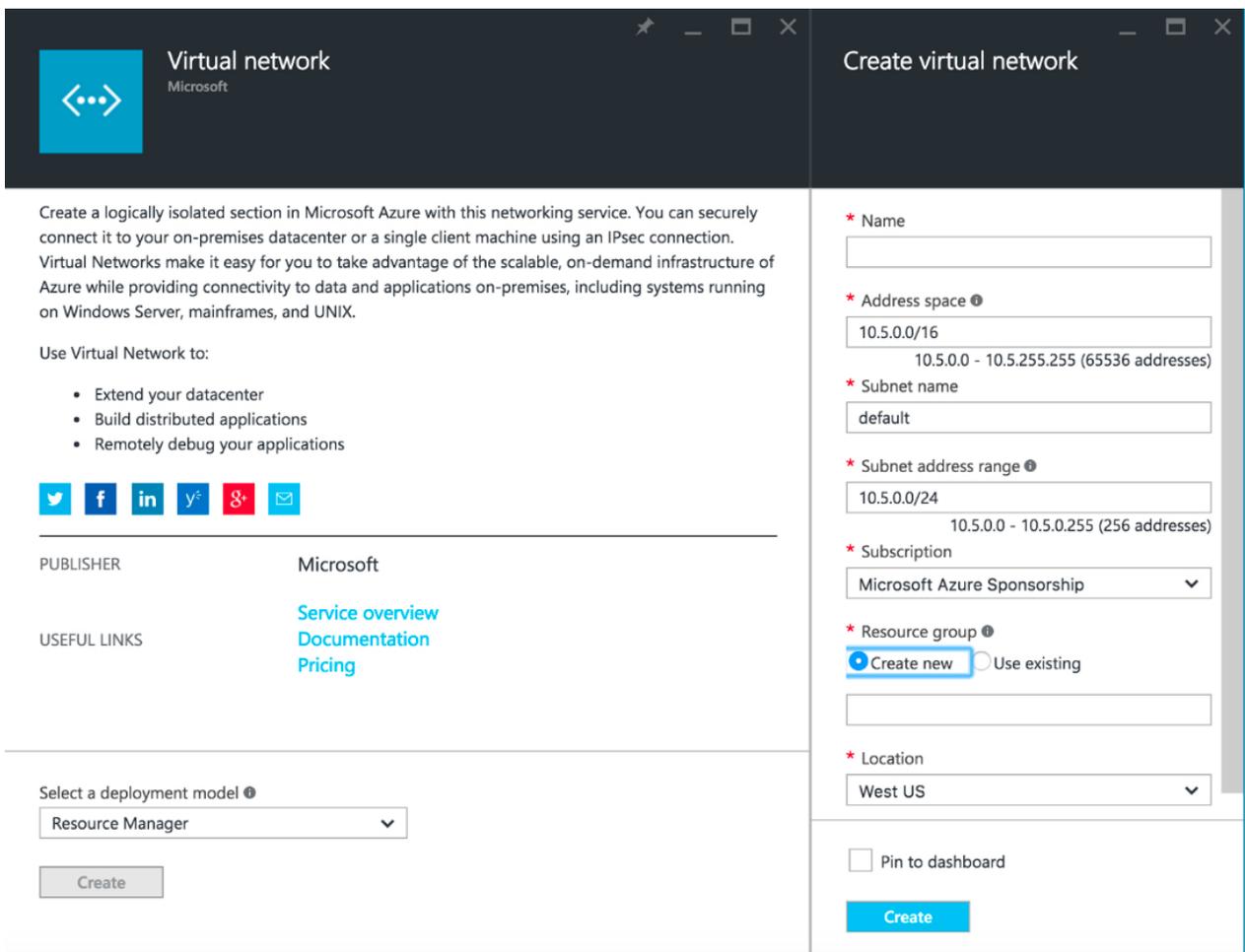
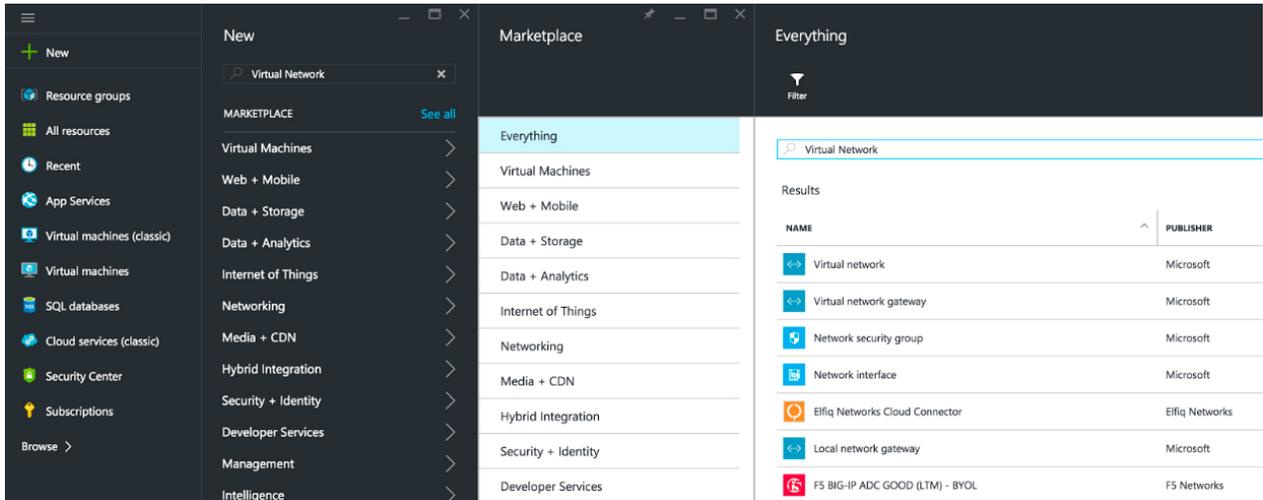
Setting Up a Virtual Network (VNet) and Subnet

Altus 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 67.

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. Pick between a managed (aligned) or unmanaged (classic) Availability Set. A managed Availability Set supports the use of Managed Disks; an unmanaged Availability Set supports Unmanaged Disks in Storage Accounts. For more information on Availability Sets, see [Manage the availability of Windows virtual machines in Azure](#) in the Microsoft Azure documentation.
9. Click **Create**.

After creating the Availability Set for master nodes, repeat the steps to create an Availability Set for worker nodes.

Create availability set
☐ ✕

*** Name**

*** Subscription**

Microsoft Azure Sponsorship(Converted to EA)
▼

*** Resource group** ⓘ

Create new Use existing

*** Location**

East US
▼

Fault domains ⓘ

3

Update domains ⓘ

20

Use managed disks ⓘ

No (Classic)

Yes (Aligned)

Pin to dashboard

Create

Automation options

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:

Getting Started with Altus Director

- 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 Altus Director Host

Creating a DNS Server and Altus Director Host

This example shows how to set up the DNS server and Altus 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 Altus 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 Altus Director:

Host FQDN suffix * ?

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
```

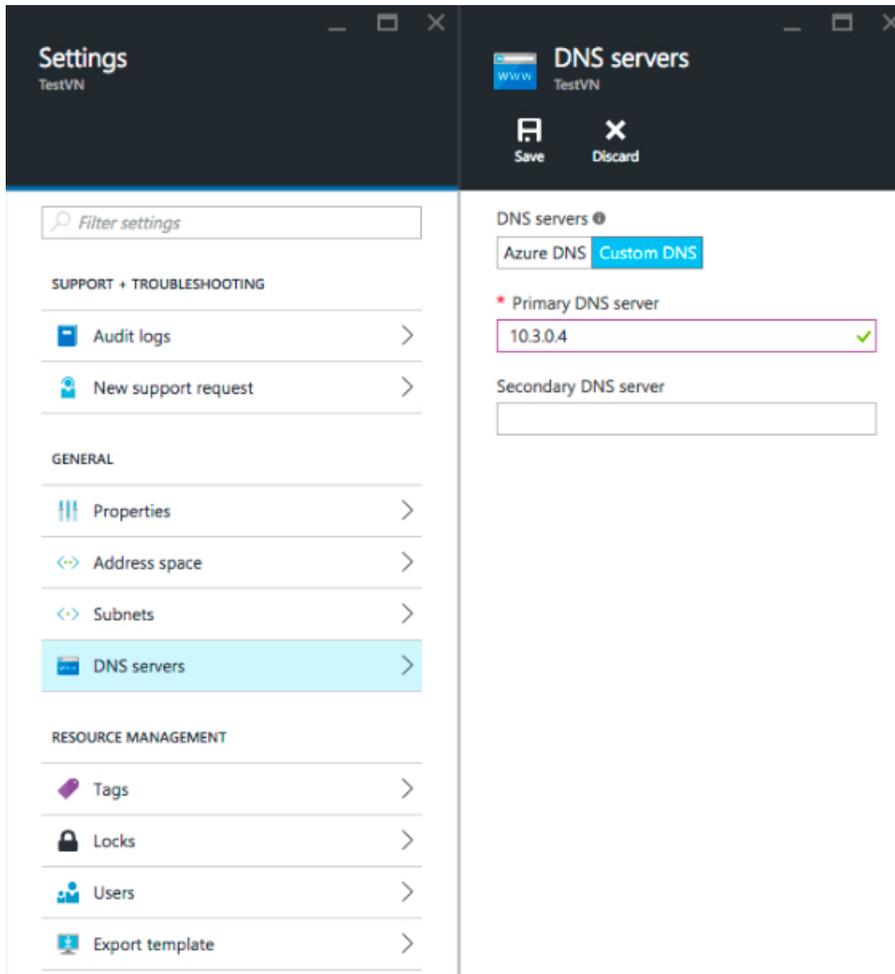
3. Set BIND to start on startup:

```
# 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 Altus 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 might 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
# CentOS and RHEL 6 use dhclient. Add a script to be automatically invoked when interface
# comes up.
# cat a here-doc representation of the hooks to the appropriate file

cat > /etc/dhcp/dhclient-exit-hooks <<"EOF"
#!/bin/bash
```

```

printf "\ndhclient-exit-hooks running...\n\treason:%s\n\tinterface:%s\n" "${reason:?}"
"${interface:?}"
# only execute on the primary nic
if [ "$interface" != "eth0" ]
then
    exit 0;
fi
# when we have a new IP, perform nsupdate
if [ "$reason" = BOUND ] || [ "$reason" = RENEW ] || [ "$reason" = REBIND ] || [ "$reason"
= REBOOT ]
then
    printf "\tnew_ip_address:%s\n" "${new_ip_address:?}"
    host=$(hostname -s)
    domain=$(nslookup $(grep -i nameserver /etc/resolv.conf | cut -d ' ' -f 2) | grep
-i name | cut -d ' ' -f 3 | cut -d '.' -f 2- | rev | cut -c 2- | rev)
    IFS='.' read -ra ipparts <<< "$new_ip_address"
    ptrrec="$(printf %s "$new_ip_address." | tac -s.)in-addr.arpa"
    nsupdatecmds=$(mktemp -t nsupdate.XXXXXXXXXX)
    resolvconfupdate=$(mktemp -t resolvconfupdate.XXXXXXXXXX)
    echo updating resolv.conf
    grep -iv "search" /etc/resolv.conf > "$resolvconfupdate"
    echo "search $domain" >> "$resolvconfupdate"
    cat "$resolvconfupdate" > /etc/resolv.conf
    echo "Attempting to register $host.$domain and $ptrrec"
    {
        echo "update delete $host.$domain a"
        echo "update add $host.$domain 600 a $new_ip_address"
        echo "send"
        echo "update delete $ptrrec ptr"
        echo "update add $ptrrec 600 ptr $host.$domain"
        echo "send"
    } > "$nsupdatecmds"
    nsupdate "$nsupdatecmds"
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
# CentOS and RHEL 7 use NetworkManager. Add a script to be automatically invoked when
interface comes up.
# cat a here-doc representation of the hooks to the appropriate file

cat > /etc/NetworkManager/dispatcher.d/12-register-dns <<"EOF"
#!/bin/bash
# NetworkManager Dispatch script
# Deployed by Altus 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=$(hostname -s)
domain=$(nslookup $(grep -i nameserver /etc/resolv.conf | cut -d ' ' -f 2) | grep -i
name | cut -d ' ' -f 3 | cut -d '.' -f 2- | rev | cut -c 2- | rev)
IFS='.' read -ra ipparts <<< "$new_ip_address"

```

```
ptrrec="$(printf %s "$new_ip_address." | tac -s.)in-addr.arpa"
nsupdatecmds=$(mktemp -t nsupdate.XXXXXXXXXX)
resolvconfupdate=$(mktemp -t resolvconfupdate.XXXXXXXXXX)
echo updating resolv.conf
grep -iv "search" /etc/resolv.conf > "$resolvconfupdate"
echo "search $domain" >> "$resolvconfupdate"
cat "$resolvconfupdate" > /etc/resolv.conf
echo "Attempting to register $host.$domain and $ptrrec"
{
  echo "update delete $host.$domain a"
  echo "update add $host.$domain 600 a $new_ip_address"
  echo "send"
  echo "update delete $ptrrec ptr"
  echo "update add $ptrrec 600 ptr $host.$domain"
  echo "send"
} > "$nsupdatecmds"
nsupdate "$nsupdatecmds"
exit 0;
EOF
chmod 755 /etc/NetworkManager/dispatcher.d/12-register-dns
service network restart
```

Checking DNS

Azure has hooks to automatically overwrite `/etc/resolv.conf` with Azure-specific values. However, depending on OS, the contents of `/etc/dhcp/dhclient-exit-hooks` or `/etc/NetworkManager/dispatcher.d/12-register-dns` are executed after 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 Altus Director and DNS, or you can add your database server to a different host in the same virtual network as Altus Director and the cluster. The supported databases are MySQL and PostgreSQL.

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

Database Server Requirements

You can install a database server on the same host as the Altus Director server, a different host in the same Virtual Network as Altus Director and the cluster, or configure an existing database. Whichever method you choose, these are requirements for the database server:

- It must be JDBC accessible both locally and remotely.
- The credentials provided to Altus 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 Installation

For MySQL, follow the instructions in [Using MySQL for Altus Director Server](#). Use the instructions for your specific version of MySQL, and keep in mind these additional 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 Altus Director Server

Altus 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 Altus 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 Altus Director Server and Client on Azure

To install Altus 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 Altus Director.
2. Install a supported version of the Oracle Java Development Kit (JDK) on the Altus Director host. Altus Director 6 supports JDK version 8.



Note: Spark 2.2 and higher requires JDK 8 and Python 2.7 or higher.

For download and installation information, see [Java SE Downloads](#).

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

Getting Started with Altus Director

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

```
cd /etc/yum.repos.d/  
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat7/cloudera-director.repo"
```

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

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

5. Start the Altus 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 Altus Director.
2. Install a supported version of the Oracle Java Development Kit (JDK) on the Altus Director host. Altus Director 6 supports JDK version 8. For download and installation information, see [Java SE Downloads](#).

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

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

```
cd /etc/yum.repos.d/  
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat6/cloudera-director.repo"
```

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

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

5. Start the Altus 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 Altus Director CLI.

- [azure.simple.conf](#): A simple Altus Director configuration that creates a Cloudera Manager node and a four-node cluster (one master and three workers).
- [azure.reference.conf](#): A reference Altus Director configuration that creates an eight-node cluster (three masters and five workers) with high availability (HA) enabled.
- [azure.kerberos.conf](#): The same Altus 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 Altus 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 Altus Director server using a private IP. You can also proxy directly to the Altus 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 Altus 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

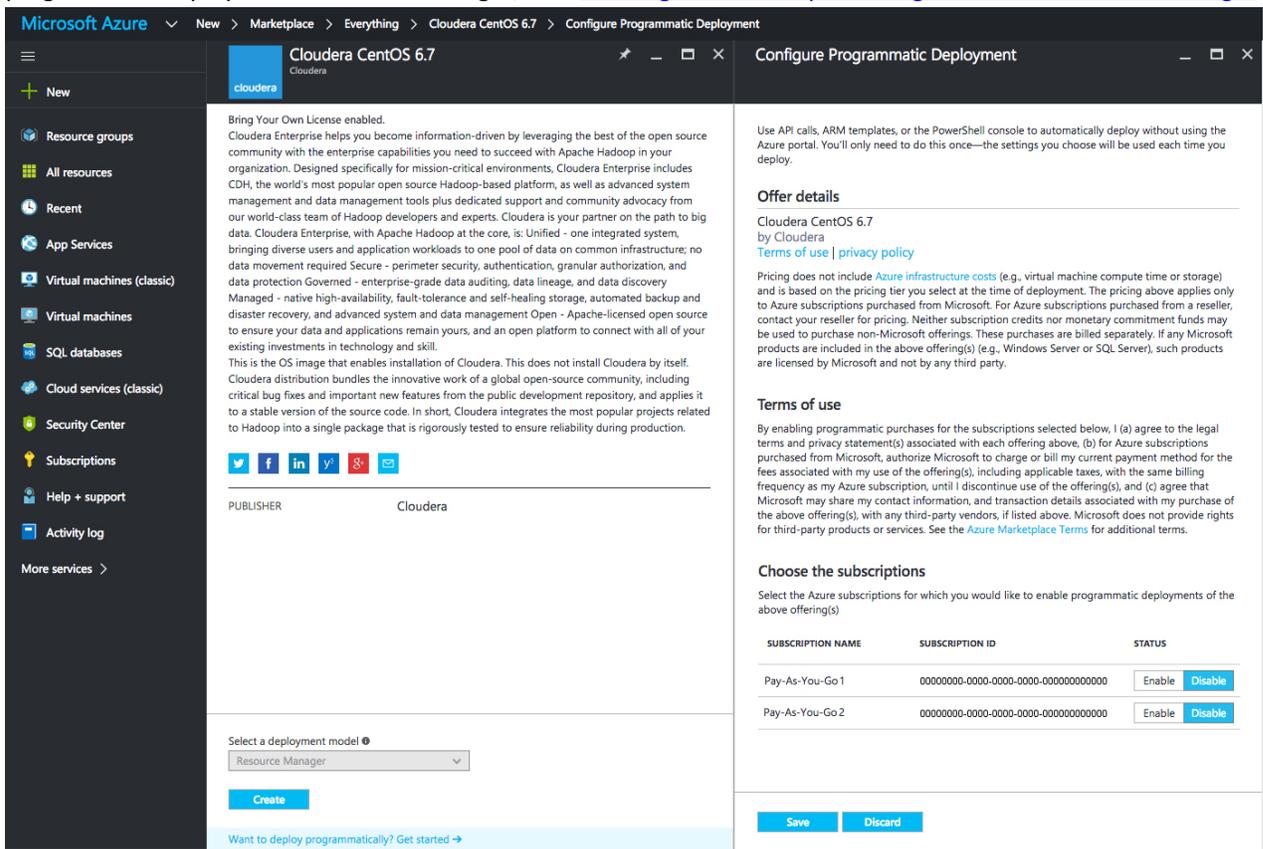
```
"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" ^
--user-data-dir="%USERPROFILE%\chrome-with-proxy" ^
--proxy-server="socks5://localhost:1080"
```

In this Chrome session, you can connect to any Altus Director–accessible host using the private IP address or internal FQDN. For example, if you proxy to the Altus Director server, you can connect to Altus Director as if it were local by entering localhost:7189 in the Chrome URL bar.

Allowing Access to VM Images

The Altus 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 Altus Director Azure plugin uses the Cloudera-certified CentOS 6 image. For detailed steps allowing programmatic deployment of Azure VM images, see [a Working with Marketplace Images on Azure Resource Manager](#).



Adding an Altus Director Environment on 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 Altus Director uses with Azure. While creating an environment, you are also prompted to deploy its first cluster.

In the **Add Environment** screen:

General Information

Environment name *	<input type="text"/>	?
Cloud provider	<input type="text" value="Microsoft Azure"/>	?
Azure Cloud Environment	<input type="text" value="azure"/>	?
Subscription ID *	<input type="text"/>	?
Tenant ID *	<input type="text"/>	?
Client ID *	<input type="text"/>	?
Client Secret *	<input type="text"/>	?

Azure

Region	<input type="text" value="westus"/>	?
--------	-------------------------------------	-------------------

SSH credentials

Username *	<input type="text"/>	?
Private key *	<input checked="" type="radio"/> File Upload <input type="radio"/> Direct Input	?
	<input type="text"/> <input type="button" value="Choose File"/>	

1. Enter a name in the **Environment Name** field.
2. In the **Cloud provider** field, select **Azure Cloud Platform**.
3. In the **Azure Cloud Environment** field, select which Azure Cloud to use.
4. In the **Subscription ID** field, enter the Azure subscription ID.
5. In the **Tenant ID** field, enter the ID of your ADD tenant. See Obtain [Obtaining Credentials for Altus Director](#) for details on obtaining the AAD tenant ID.
6. In the **Client ID** field, enter the client ID of the Azure service principal you created earlier. See [Obtaining Credentials for Altus Director](#) for details on obtaining the client ID.
7. In the **Client Secret** field, enter the client secret of the Azure service principal you created earlier. See [Obtaining Credentials for Altus Director](#) for details on obtaining the client secret.
8. In the **Region** field, select which location to use.
9. 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. Altus Director uses the SSH key pairs to create and access VMs in Azure.

10 Click **Continue** to add the environment and advance to creating a cluster, either with the simple setup procedure or the advanced setup procedure:

Environment successfully created. ✕

Simple Setup

Use default settings to create a cluster.

Advanced Setup

Define advanced settings and topology.

A **Creating a cluster with simple setup** is quick and easy because many configuration choices have been made for you. This is a great way to try out the product, and get a cluster up and running quickly. The **advanced setup** procedure gives you many more options for how you want your cluster to be configured, and can be used to set up advanced features, like Kerberos, TLS, and external databases.

To proceed, and create one of these kinds of clusters, choose one of the following topics:

- [Simple Setup: Creating an Azure Cluster with Default Settings](#) on page 80
- [Advanced Setup: Installing Cloudera Manager and CDH on Azure](#) on page 82

Simple Setup: Creating an Azure Cluster with Default Settings

Simple setup clusters provide the simplest and most reliable way to get a Cloudera Manager deployment and CDH cluster up and running quickly when you do not require custom configurations or advanced features, like Kerberos, TLS, or external databases. To bootstrap a new cluster, you need only provide some information about your cloud provider of choice, and about the type of cluster you want to create. All of the other details about how your cluster is configured, like its topology and versions of Cloudera Manager and CDH, are determined for you by Altus Director.

Simple setup clusters are not recommended as production clusters. But a good strategy for creating a production cluster is to export a client configuration file for a simple setup cluster, and then edit that configuration file to add more advanced features. In this way, you simplify the task of creating a custom cluster, beginning from a known working configuration, rather than starting from the more complete, but more complex [azure.reference.conf](#) file. You can export a client configuration file through the web UI or through the server API. For more information, see [Exporting a Configuration File](#) on page 201

Type of Simple Setup Clusters

You can choose from five different types of cluster with the simple setup procedure, based on the workloads you will run in the cluster. The workload type you choose determines the services in the cluster.

- **Basic:** Provides a simple Hadoop environment. Includes Hive and MapReduce.
- **Data Engineering:** For Spark workloads and ETL jobs. Includes Hive and Spark.
- **Analytic Database:** For business intelligence and SQL analytics. Includes Impala and Hive.
- **Operational Database:** For NoSQL real-time application serving. Includes HBase.
- **Enterprise Data Hub:** Provides a comprehensive range of services for the Cloudera platform.

Ways to Create a Simple Setup Cluster

There are several ways to create a simple setup cluster:

- Use the Altus Director UI, and clicking the tile **Simple Setup** at the conclusion of the **Add Environment** procedure, or by navigating to an existing environment and clicking the **Add Cluster** button.
- Use the Altus Director CLI and the `bootstrap-remote` command with a configuration file. You can define a simple client configuration file with the same environment and cluster information that you would supply through the web user interface. For a sample configuration file, including instructions for creating a cluster, see [azure.simple_setup.conf](#) on the Cloudera GitHub site.
- Submit JSON or HOCON input to the Altus Director API import endpoint. For details, see `importClientConfig` in the Altus Director API console at `director_ip_address:port/api-console/index.html`.

Launching a Simple Setup Cluster with the Altus Director UI

To use simple setup to launch a cluster with the Altus Director UI, perform the following steps:

1. Add an Altus Director environment, as described in [Adding an Altus Director Environment on Azure](#) on page 78.
2. Begin the simple setup installation procedure in one of the following ways:
 - Click the **Simple Setup** tile at the conclusion of the **Add Environment** procedure.
 - Navigate to an existing environment and click the **Add Cluster** button.
3. Provide values for the fields on the **Add Cluster** screen:
 - a. **Environment name:** The name of the Environment that includes this cluster. [More information.](#)
 - b. **Cluster name:** Between 2 and 40 alphanumeric characters. Space, underscore, and hyphen are allowed except at the beginning or end of the name. Azure instance names for the cluster nodes will be prefixed with the cluster name.
 - c. **Workload Type:** Select the typical workload that this cluster will run on from the dropdown list. The workload type will be used to determine the services that will run inside the cluster.
 - d. **Worker Node Instance Type:** The machine type. [More information.](#)
 - e. **Worker Node Count:** Number of cluster nodes for worker roles.
 - f. **Virtual Network:** The Virtual Network for the deployment. This must exist in the Virtual Network Resource Group you selected. [More information.](#)
 - g. **Virtual Network Resource Group:** The Resource Group where the Virtual Network is located.
 - h. **Network Security Group:** The Network Security Group for the deployment. This must exist in the Network Security Group Resource Group you selected. [More information.](#)
 - i. **Network Security Group Resource Group:** The Resource Group where the Network Security Group is located.
 - j. **Compute Resource Group:** The Resource Group where the compute resources such as VM instances and availability sets will be created. [More information.](#)
 - k. **Host FQDN suffix:** The private domain name used to create a FQDN for each host. Note: Azure provided DNS service does not support reverse DNS lookup for private IP addresses. You must setup a dedicated DNS service with reverse lookup support and use this FQDN Suffix.
4. Click **Continue** to launch the cluster.

If you launch additional simple setup clusters using the same Altus Director environment, many of the fields on the **Add Cluster** screen will be pre-populated with the values you used for your previous simple setup cluster.

Advanced Setup: Installing Cloudera Manager and CDH on Azure

Before You Deploy Cloudera Manager and CDH



Important: Before using Altus 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. For more information see [Allowing Access to VM Images](#) on page 78.

This topic describes how to set up Cloudera Manager and a CDH cluster in Microsoft Azure using the Altus 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.
- Altus Director server VM.
- Altus Director server installed and running.
- Altus 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

This section describes how to create a cluster using the advanced setup procedure in the Altus Director UI. For the simple setup procedure, see [Simple Setup: Creating an Azure Cluster with Default Settings](#) on page 80.

To deploy Cloudera Manager and CDH on an Azure VM instance, begin by creating an environment as described in [Adding an Altus Director Environment on Azure](#) on page 78. The environment defines common settings, like region and key pair, that Altus Director uses with Azure.

To deploy Cloudera Manager and launch CDH clusters on Azure, perform the following steps:

1. Open a web browser and go to the private IP address of the instance you created running Altus Director server. Include port 7189 in the address, for example: `http://192.0.2.0:7189`.
2. In the Altus Director login screen, enter `admin` in both the **Username** and the **Password** fields.
3. In the Altus Director **Welcome** screen, click **Let's get started**. This opens a wizard for adding an environment, Cloudera Manager, and a CDH cluster.
4. Click **Continue** to add Cloudera Manager.
5. 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.
6. 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** field, select one of the available images or use this format to define any Azure Marketplace VM image inline (replace the italicized values with the actual names for the publisher, offer, sku, and version):

```
/publisher/publisher/offer/offer/sku/sku/version/version
```

- 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. Note that there are two types of Availability Sets: managed (aligned) and unmanaged (classic). For more information see [How to Use Availability Sets](#) in the Microsoft Azure documentation.
- n. In the **Use Managed Disks** field, select **Yes** to use Managed Disks, or **No** to use Storage Accounts. Note the following considerations:
- Azure Managed Disks are an improvement over Azure Storage Accounts. They simplify storage, provide better reliability for [availability sets](#), and improve scalability. See [Azure Managed Disks Overview](#) in the Microsoft Azure documentation for more information.
 - A VM with Managed Disks can only be deployed into a managed Availability Set. For more information see [How to Use Availability Sets](#) in the Microsoft Azure documentation.
- o. In the **Instance name prefix** field under **Advanced Options**, enter the desired instance name prefix.
- p. In the **Storage Account Type** field, select **Premium_LRS**. For instance templates intended for worker nodes, you can select **Standard_LRS**. See the [Cloudera Reference Architecture for Microsoft Azure Deployments](#) for details on supported storage account types and configurations.
- q. In the **Data Disk Count** field in **Advanced Options**, enter the number of data disks to attach for the VM.
- r. In the **Data Disk Size in GiB** field, leave the value at **1024** or pick from the dropdown of available sizes.
- s. Leave the **SSH username** field blank to use the username you set when you created the environment at [step 9.a](#) on the page [Adding an Altus Director Environment on Azure](#) on page 78.
- t. 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.

7. 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 1, 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.

Licensing

Desired License Type * Cloudera Enterprise ?

Please provide a Cloudera Manager license key.

License Key * File Upload Direct Input ?

Choose File

Billing ID ?

To enable usage-based billing, you must have a Cloudera Enterprise license and a billing ID provided by Cloudera. 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.

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

Altus Director version	Cloudera Manager version installed
Altus Director 2.0	Latest released version of Cloudera Manager 5.5
Altus Director 2.1	Latest released version of Cloudera Manager 5.7
Altus Director 2.2	Latest released version of Cloudera Manager 5.8
Altus Director 2.3	Latest released version of Cloudera Manager 5.10
Altus Director 2.4	Latest released version of Cloudera Manager 5.11
Altus Director 2.5	Latest released version of Cloudera Manager 5.12
Altus Director 2.6	Latest released version of Cloudera Manager 5.13
Altus Director 2.7	Latest released version of Cloudera Manager 5.14
Altus Director 2.8	Latest released version of Cloudera Manager 5.15
Altus Director 6.0	Latest released version of Cloudera Manager 6.0

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 <https://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 https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.5.4/.



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

- c. In the **Repository Key URL** field, enter the URL for the repository key. Repository key URLs have the same form as repository URLs except they end with the name of the key file instead of the Cloudera Manager version. For example, the repository key URL for any version of Cloudera Manager 5 on any supported version of Red Hat 7 is https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera.

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

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

11. 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 Altus Director you are using:

Altus Director version	CDH version installed
Altus Director 2.0	Latest released version of CDH 5.5
Altus Director 2.1	Latest released version of CDH 5.7
Altus Director 2.2	Latest released version of CDH 5.9
Altus Director 2.3	Latest released version of CDH 5.10
Altus Director 2.4	Latest released version of CDH 5.11
Altus Director 2.5	Latest released version of CDH 5.12
Altus Director 2.6	Latest released version of CDH 5.13
Altus Director 2.7	Latest released version of CDH 5.14
Altus Director 2.8	Latest released version of CDH 5.15
Altus Director 6.0	Latest released version of CDH 6.0

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 <https://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 <https://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.

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

Add Group

12 Click **Continue**.

13 At the confirmation prompt, click **OK** to deploy the cluster. Altus Director displays a status screen.

Status

TESTCLUSTER01 Bootstrapping

7 / 30

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

1. Starting
2. Starting
3. Starting

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

Terminating an Azure Deployment

How to Terminate an Azure Deployment

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

1. In Altus 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 Altus Director (the configuration file or database, for example), back it up.
3. In the Azure web UI, terminate the Altus Director instance and any other instance Altus Director was unable to terminate.
4. If applicable, terminate any external database you configured Altus Director to use.

Adding New VM Images, Custom VM Images, Regions, and Instances

The Altus Director Azure Plugin supports adding new VM images, regions, and instances by modifying configuration files. For more information see [Altus 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

Azure U.S. Government is supported with no additional configuration needed. Select **azure-us-government** in the Azure Cloud Environment field when adding a new environment.

Configuring and Deploying to Azure Germany Regions

Azure Germany is supported with no additional steps. Select **azure-germany** in the Azure Cloud Environment field when adding a new environment.



Note: For Azure Germany deployments, the JDK for Altus Director must be Java SE 8 Update 101 (8u101) or higher so that the CA root certificate authority called D-TRUST (used by Azure Germany API endpoints) is supported.

Deploying Clusters with Custom Images



Note: Cludera recommends that you use one of the [Cludera CentOS Virtual Machine images](#) as the base for your custom images.

Deploying custom images is supported by updating instance template fields:

1. Set the **Image** field to be the Resource Id for the custom image. The Resource Id is on the Custom Image's **Overview** pane and is in the format (replace italicized words with your actual values):

```
/subscriptions/subscription-id/resourceGroups/resource-group-name/providers/Microsoft.Compute/images/custom-image-name
```



Note: If you are using a configuration file, this field is named `image`.

2. Set the **Use Custom Managed VM Image** field to **Yes**. The custom image option is only supported with Managed Disks.



Note: If you are using a configuration file, this field is named `useCustomManagedImage`.

3. Set the **Custom VM Image purchase plan** field in this format (replace italicized words with your actual values):

```
/publisher/value/product/value/name/value
```

If there's no plan leave the field blank.



Note: If you are using a configuration file, this field is named `customImagePlan`.

4. If the custom image has a data disk attached, then set **dataDiskCount** to 0. If you just comment it out, it will default to 5.



Note: **dataDiskCount** is found in the instance template section of the Altus Director configuration file, and on the [Create New Instance Template screen](#) of the UI.

Important Notes About Altus 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 Altus Director.

Not all Azure Virtual Machine (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 [Cludera Reference Architecture for Microsoft Azure Deployments](#) for the latest supported VM types.

Getting Started with Altus Director

Azure Resources Managed by Altus Director

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

- Managed disks and storage accounts:
 - For VMs using unmanaged disks in storage accounts: one storage account for each VM.
 - For VMs using managed disks: one managed disk for each data disk, and one managed disk for the OS disk.
- A NIC for each VM.
- A public IP address for each VM, if public IP addresses are enabled.

Changing the Instance Type is Not Supported in Azure

- Changing the instance type of an already-deployed VM is not supported in Azure.
- Changing the instance type of a VM through a tool external to Altus Director is not supported in Azure. You cannot, for example, use the Azure Portal to change the instance type. Altus Director is not updated when instance type changes are made with external tools, and your cluster will show errors in Altus Director.

Deploying Production Clusters

Although the Altus 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 Altus 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 Altus Director operations such as `allocate` to fail prematurely. If this happens, you might 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 [Altus 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.
4. On the Altus Director server, copy the modified `azure-plugin.conf` to `/var/lib/cloudera-director-plugins/azure-provider-x.x.x/etc/azure-plugin.conf` (replacing `x.x.x` with the latest version), and then restart Altus Director with `sudo service cloudera-director-server restart`

This procedure changes only the Azure plug-in timeout. The following Altus 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 Altus 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 used for the VM OS disk and cluster data disks.
- 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.
- If the VM was set up to have a public IP, Altus Director will delete it. If a public IP was attached manually after the VM was created, Altus Director will not delete it.



Important: Based on the deletion behaviors described, do not reuse any resources created by the Azure plug-in for any other purpose.

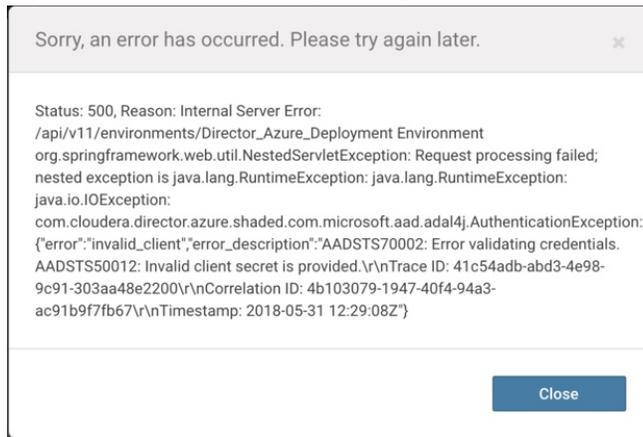
Updating an Expired Azure Client Secret

The Azure client secret is a necessary part of Microsoft Azure security. Altus Director uses the combination of your Azure Active Directory application tenant ID, client ID, and client secret to authenticate, and thereby perform actions against Azure.



Note: For more information, see [Use portal to create an Azure Active Directory application and service principal that can access resources](#).

If your client secret expires, you will get an error in the Altus Director UI similar to the following:



An expired client secret also results in an entry similar to the following in the Altus Director log file:

```
Failed to authenticate with Azure: java.io.IOException:
com.cloudera.director.azure.shaded.com.microsoft.aad.adal4j.AuthenticationException:
{"error": "invalid_client", "error_description": "AADSTS70002: Error validating
credentials. AADSTS50012: Invalid client secret is provided.\r\nTrace ID:
fc5529da-2536-44f6-bad4-d07f6a9bbd00\r\nCorrelation ID:
1e7f12f3-32e7-4635-8b27-197e53fd0ab8\r\nTimestamp: 2018-05-31
11:13:11Z" }
```

Steps to Update an Expired Client Secret

To update an expired client secret, perform the following steps for each environment that uses the expired secret.



Note: This procedure works for Altus Director 2.6 and higher. If you are using a lower version of Altus Director, contact Cloudera Support for assistance.

1. Turn off credential validation.

- a. `ssh` to the Altus Director server host.
- b. Locate the Azure plugin configuration file if you have one, or create a new plugin configuration file.



Note: Whether an Azure plugin configuration file exists on your system will depend on how Altus Director was initially installed and whether any plugin configuration settings have already been changed.

- a. If you already have a configuration file, it will be located at
`/var/lib/cloudera-director-plugins/azure-provider-version/etc/azure-plugin.conf`.

b. If there is no `azure-plugin.conf` file at that location, create one by running the following command:

```
wget -O /var/lib/cloudera-director-plugins/azure-provider-version/etc/azure-plugin.conf  
https://raw.githubusercontent.com/cloudera/director-scripts/master/azure-plugin-config/azure-plugin.conf
```

c. Edit the `azure-plugin.conf` file. Towards the end of the file there's an `azure-validate-credentials` field. Change the value to `false`:

```
azure-validate-credentials: false
```

2. Restart Altus Director:

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

3. Update the credentials in either of the following ways:

- Using the Altus Director UI:
 1. Log into Altus Director and go to the environment whose client secret has expired.
 2. From the **Add Cloudera Manager** dropdown menu, select **Update Environment Credentials**.
 3. Put in the new client secret along with your existing settings for this environment. Existing environment settings can be found under the **Details** tab.
- Using the Altus Director API:
 1. Log into Altus Director. (You must be authenticated to use the Altus Director API.)
 2. Go to the **environments** section of the API console of your Altus Director server (replace `director-server-hostname` with your own):

```
http://director-server-hostname:7189/api-console/index.html#
```

3. Click on the `GET /api/v11/environments` section, and click **Try it out!** to list the environments. If you are using a version of the API other than `v11`, this procedure will still work, but your URL will contain the version you are using in place of `v11`.
4. Copy the name of the environment whose client secret has expired. Open the `GET /api/v11/environments/{name}` section and paste the environment name into the `name` parameter box, and click **Try it out!** to display the environment details.
5. Copy the `config` portion of the JSON block including the curly braces but excluding the `"config"` string. It should be similar to the example below. You might have different fields and values, but **clientSecret** should be there. If the **clientSecret** field is not there, check to be sure you are using the correct API.

```
{  
  "tenantId": "[...]",  
  "region": "eastus2",  
  "azureCloudEnvironment": "azure",  
  "clientSecret": "REDACTED",  
  "subscriptionId": "[...]",  
  "clientId": "[...]"  
}
```

6. Open the `PUT /api/v11/environments/{name}/provider/credentials` section. Note that there is `/provider/credentials` in the path - it is **NOT** the `PUT /api/v11/environments/{name}` section.
7. Paste in the whole JSON block, including all fields and the surrounding curly braces, and then edit the JSON block:
 - Remove the `"region"` line.
 - Replace the `"REDACTED"` portion of `"clientSecret": "REDACTED"` with your new client secret.

8. Click **Try it out!** again to update the credentials. On success, you should get a 202 response code. If you get a 400 Bad Request, check the Altus Director logs for full details. Also, check that you have not made any of the following common mistakes:

- a. Using `PUT /api/v11/environments/{name}` instead of `PUT /api/v11/environments/{name}/provider/credentials` (note the `/provider/credentials` in the path).

This causes the following response:

```
"status": 400,
"error": "Bad Request",
"exception":
"org.springframework.http.converter.HttpMessageNotReadableException",
"message": "Bad Request"
```

- b. Improperly formatted JSON. This causes the following response:

```
"status": 400,
"error": "Bad Request",
"exception":
"org.springframework.http.converter.HttpMessageNotReadableException",
"message": "Bad Request"
```

- c. Forgetting to remove the `"region"` line. This causes the following response:

```
"status": 400,
"error": "Bad Request",
"exception":
"com.cloudera.launchpad.api.common.EnvironmentsResource$UnsupportedEnvironmentUpdateException",
"message": "Unsupported environment update request"
```

4. Turn on credential validation. In this step, you undo what you did in the beginning:

- If you did not have an `azure-plugin.conf` file initially, delete the file now.
- If there was a file initially, then change `azure-validate-credentials` in your `azure-plugin.conf` back to `true`:

```
azure-validate-credentials: true
```

5. Restart Altus Director:

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

6. Verify the fix.

- a. Tail the logs on the Altus Director host:

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

- b. Log into the Altus Director UI and go to the environment whose client secret was previously expired.
- c. If there are no errors with credential validation in the Altus Director UI or in the logs, then the new client secret has been applied.

If you see the following message in the logs after restarting Altus Director, credential validation is still turned off (`azure-validate-credentials: false`):

```
Skipping Azure credential validation with Azure backend.
```

Getting Started with Altus Director

Verify that everything was changed back correctly, and that Altus Director was restarted.

Useful Links

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

Usage-Based Billing

Altus 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 Altus Director.

Prerequisites

The following are required for usage-based billing:

- Altus 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 Altus Director to deploy Cloudera Manager and CDH.
- Outbound HTTPS connectivity from Altus 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 Altus 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, Altus 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 Altus Director is stopped. Billing resumes when Altus Director is started, but the billing hour for all billable clusters is reset from when Altus 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 Altus 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 Altus Director server web UI or with the Altus Director client and the `bootstrap-remote` command, as described in this section.

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

The procedure for deploying Cloudera Manager and CDH through the Altus Director web UI is described in [Advanced Setup: Installing Cloudera Manager and CDH on AWS](#) on page 43. To enable usage-based billing, follow the procedure as described there, but be sure to provide a Cloudera Enterprise license and a billing ID as described in the steps for the **Add Cloudera Manager** screen.

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



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

Licensing

Desired License Type * ?

Please provide a Cloudera Manager license key.

License Key * File Upload Direct Input ?

Billing ID ?

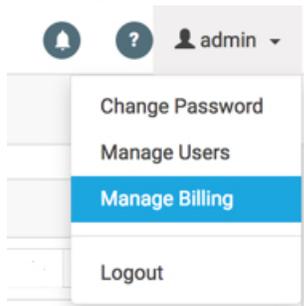
Enabling Usage-Based Billing with bootstrap-remote

The procedure for deploying Cloudera Manager and CDH through the Altus Director client using the `bootstrap-remote` command is described in [Submitting a Cluster Configuration File](#) on page 200.

There is a [sample Altus 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 Altus Director web UI.



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

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

Cloudera Manager	Environment	License Type	Billing ID ⓘ	Actions
CM01	Env01	Cloudera Enterprise	*****kSepAAA	Update Billing ID ▾

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

1. On the Manage Billing page, click **Assign Billing ID** to open the **Update Billing ID** dialog.
2. Enter a valid billing ID.
3. Click **Update**.

To replace a billing ID with a different one:

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

Troubleshooting Network Connectivity for Usage-Based Billing

If Altus 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 Altus Director web UI, and the bell icon turns red. Click the icon to see the alert.

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

Usage-Based Billing

- Altus 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 Altus 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 might have failed or might have a connectivity issue.

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

- Check that the firewall rules for your Altus 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 Altus Director instance has a route to the internet.
- Check in the Altus Director web UI to ensure that Altus Director is able to connect to the Cloudera Manager instance.
- Open a shell on the Altus 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 Altus 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 might be down. Contact Cloudera Support.

Altus Director Usage Bundles

An Altus 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 an Altus Director block.



Note: All passwords and sensitive information included in Altus 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.

Altus Director Block

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

Initial metadata in the block includes the metadata structure version and the host, port, and API version of Altus Director itself. Ensuing details begin with the version of Altus 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 Altus 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 Altus Director are listed. As with other Altus 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

Altus 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 Altus Director is sending out.

To enable usage logging in the Altus 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 Altus Director so that the change takes effect.

Usage-Based Billing

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 Altus Director instance. Do not enable it for long periods of time, to avoid running out of disk space.

Using Cloud Provider Regions

This section explains how to use cloud provider regions with Altus Director.

Running Altus Director and Cloudera Manager in Different Regions or Clouds

A Altus Director instance requires network access to all of the Cloudera Manager and CDH instances it deploys and manages. If Altus 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 Altus Director in one region and Cloudera Manager and the CDH clusters it manages in a different region.
- Installing Altus 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 Altus 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 Altus Director access to the private subnet. Alternatively, Altus Director can be given SSH access to the instances through the public internet.

When using SSH to configure Cloudera Manager and CDH instances, Altus 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 Altus 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 Altus 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 Altus Director can run in a different subnet, Cloudera Manager and the CDH cluster hosts must be in the same subnet.
- Altus Director must have SSH access to the public IP addresses of all cluster instances.
- Altus 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 Altus Director, if Altus 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 Altus Director

Altus 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 Altus Director releases. But, because you might want to use a new region before it has been added in a new release, Altus 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, Altus 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 Altus 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, Altus 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 might 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 Altus Director's AWS plugin, so it will be used for all environments created with that instance of Altus 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 Altus Director releases, as the plugin changes version. For example, `aws-provider-1.4.1` matches with Altus Director 2.4. So the path and file name with Altus Director 2.4 would be as follows:

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

Altus 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 Altus 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

Altus 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 Altus Director. Altus 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 might 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

Configuring Storage for Altus Director

This section explains options for configuring storage for Altus Director. A number of kinds of storage and storage options are available with Altus Director. You can use an external database for Altus Director server in place of the embedded H2 database, and doing so is highly recommended for production installations. Altus Director supports MySQL and MariaDB for this purpose. Also, you can configure Altus Director to encrypt the data stored in the Altus Director database. These options are described in the following sections.

Using MySQL for Altus Director Server



Note: This section is about the data Altus 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 124.



Note: For information on configuring Altus Director and the MySQL database to communicate using TLS, see [Enabling TLS with Altus Director](#), and especially the section, [TLS for Altus Director's Database](#) on page 150.

Altus 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 102 to verify that your MySQL configuration meets the requirements for Altus 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 and Centos	\$ sudo yum install mysql-server
Ubuntu	\$ sudo apt-get install mysql-server

After issuing the command, you might 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 and Centos	\$ sudo service mysqld stop
Ubuntu	\$ 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 Altus 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
```

Configuring Storage for Altus Director

```
[mysqld_safe]
log-error=/var/log/mysql.log
pid-file=/var/run/mysql/mysql.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 and Centos	<pre>\$ sudo /sbin/chkconfig mysqld on \$ sudo /sbin/chkconfig --list mysqld mysqld 0:off 1:off 2:on 3:on 4:on 5:on 6:off</pre>
Ubuntu	<pre>\$ sudo chkconfig mysql on</pre> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p> Note: <code>chkconfig</code> might not be available on recent Ubuntu releases. You might need to use Upstart to configure MySQL to start automatically when the system boots. For more information, see the Ubuntu documentation or the Upstart Cookbook.</p></div>

7. Start the MySQL server:

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

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

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

Installing the MySQL JDBC Driver

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

OS	Command
RHEL and Centos	<ol style="list-style-type: none">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:<pre>tar zxvf mysql-connector-java-version.tar.gz</pre>

OS	Command
	<p>3. Add the JDBC driver, renamed, to the relevant server. For example:</p> <pre data-bbox="695 254 1472 394"> \$ sudo cp mysql-connector-java-version/ mysql-connector-java-version-bin.jar /usr/share/java/ mysql-connector-java.jar </pre> <p>If the target directory does not yet exist on this host, you can create it before copying the JAR file. For example:</p> <pre data-bbox="695 510 1472 667"> \$ sudo mkdir -p /usr/share/java/ \$ sudo cp mysql-connector-java-version/ mysql-connector-java-version-bin.jar /usr/share/java/ mysql-connector-java.jar </pre> <div data-bbox="678 705 1425 873" style="border: 1px solid green; padding: 5px;"> <p> Note: Do not use the <code>yum install</code> command to install the MySQL connector package, because it installs the openJDK, and then uses the Linux <code>alternatives</code> command to set the system JDK to be the openJDK.</p> </div>
Ubuntu	<code>\$ sudo apt-get install libmysql-java</code>

Creating a Database for Altus Director Server

You can create the database on the host where the Altus Director server will run, or on another host that is accessible by the Altus 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. Altus 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 Altus 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 Altus Director configuration settings described below in [Configure Altus 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

```

You can restore the MySQL database from your backed-up copy.



Note: Be sure to stop Altus Director server before backing up or restoring. For backups, it is important that no database writes are happening while the backup is being made. For restoring, Altus Director server should be stopped because it won't reread its database until you restart it.

Configuring Altus Director Server to use the MySQL Database

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



Note: If the Altus 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 Altus Director Server from H2 to MySQL

If you are already running Altus 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. Altus 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 Altus Director requires.



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

Step 1: Prepare databases

1. Stop Altus 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 Altus 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 Altus 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.

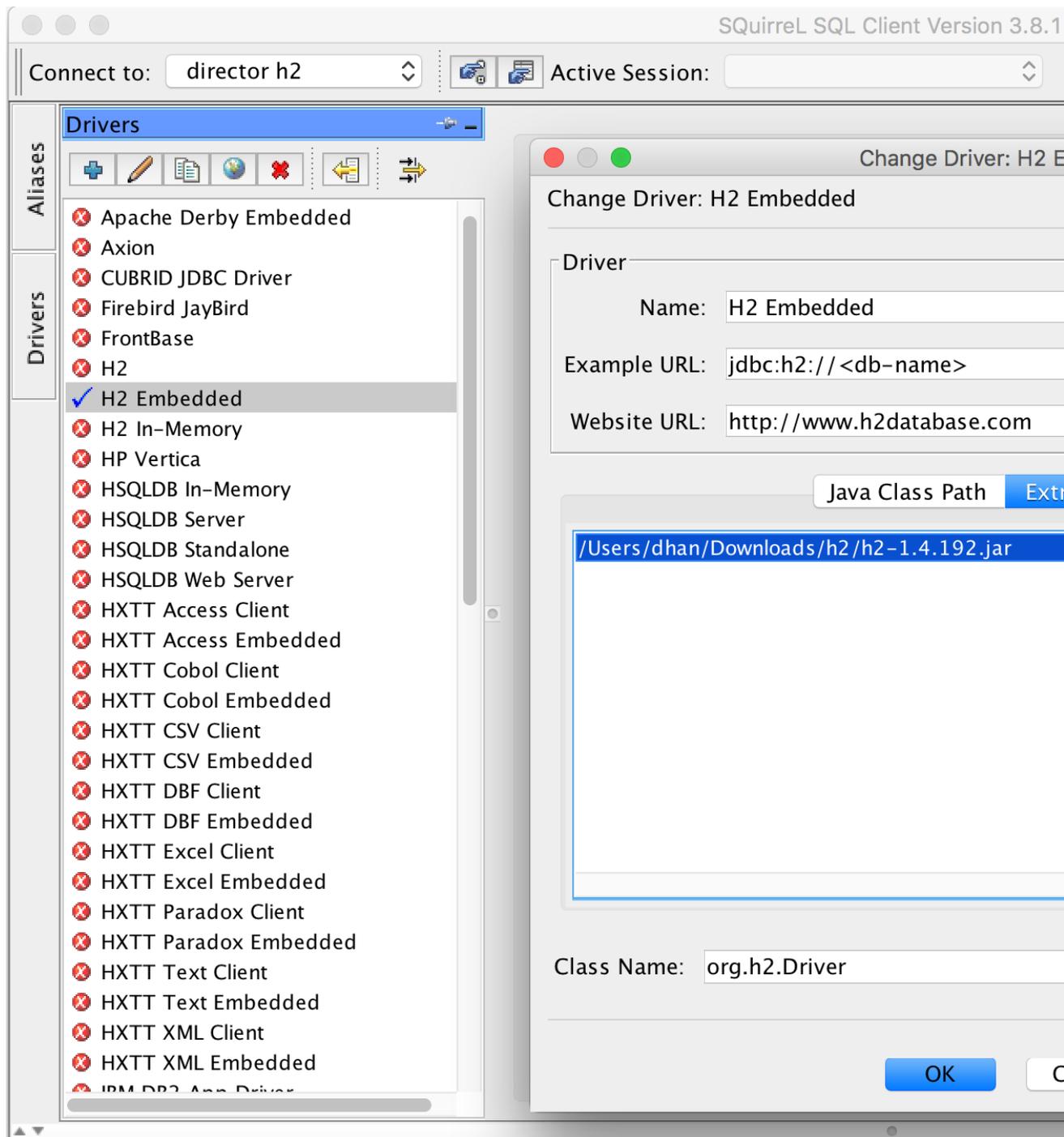


Note: H2, MySQL, and SQuirreL SQL can be run on different nodes.

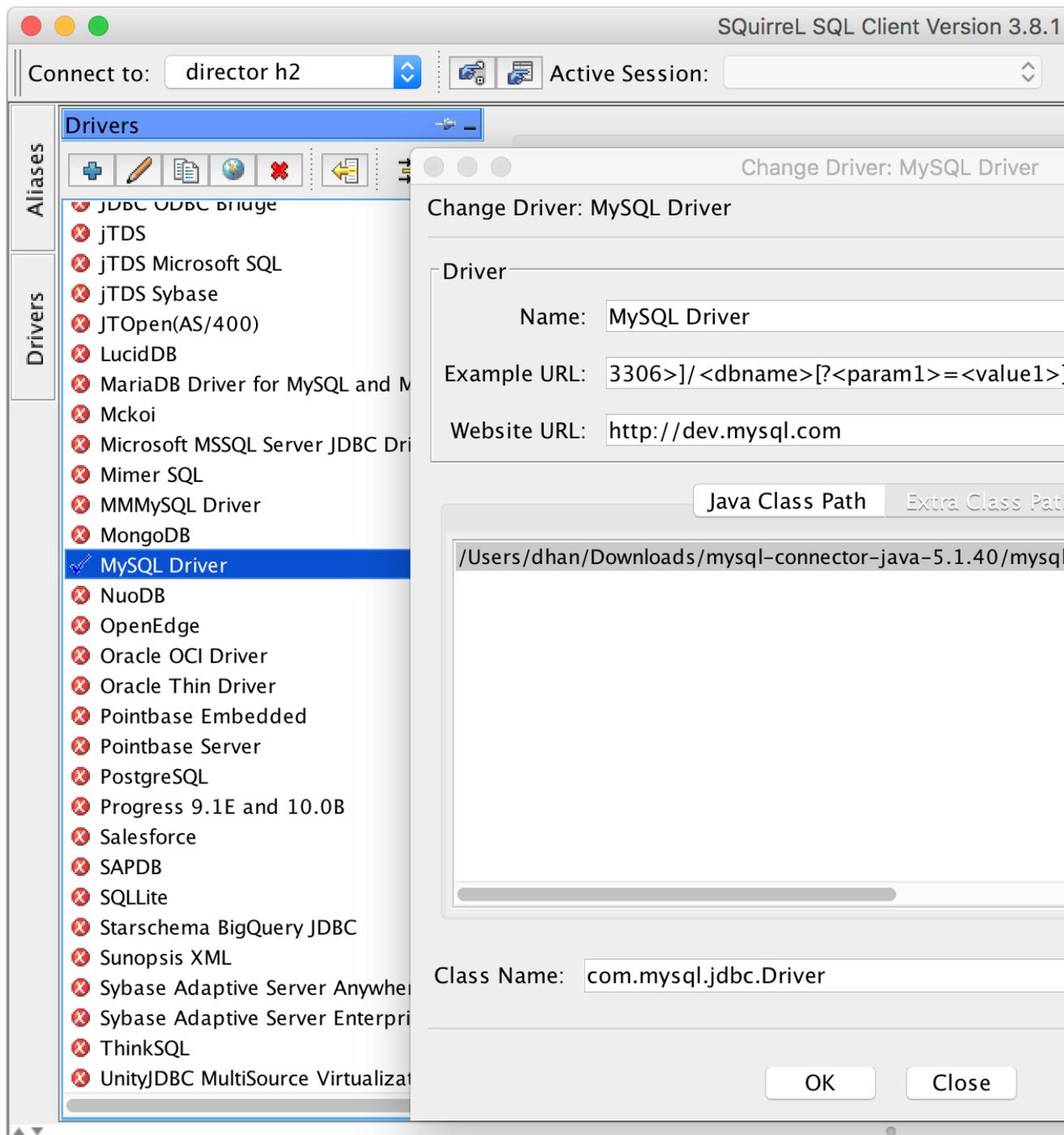
- a. Download SQuirreL SQL from the [SQuirreL SQL](#) web site.
- b. Install SQuirreL SQL with the H2 and MySQL database plugins enabled.

2. Enable the **H2 Embedded** driver in SQuirreL SQL.

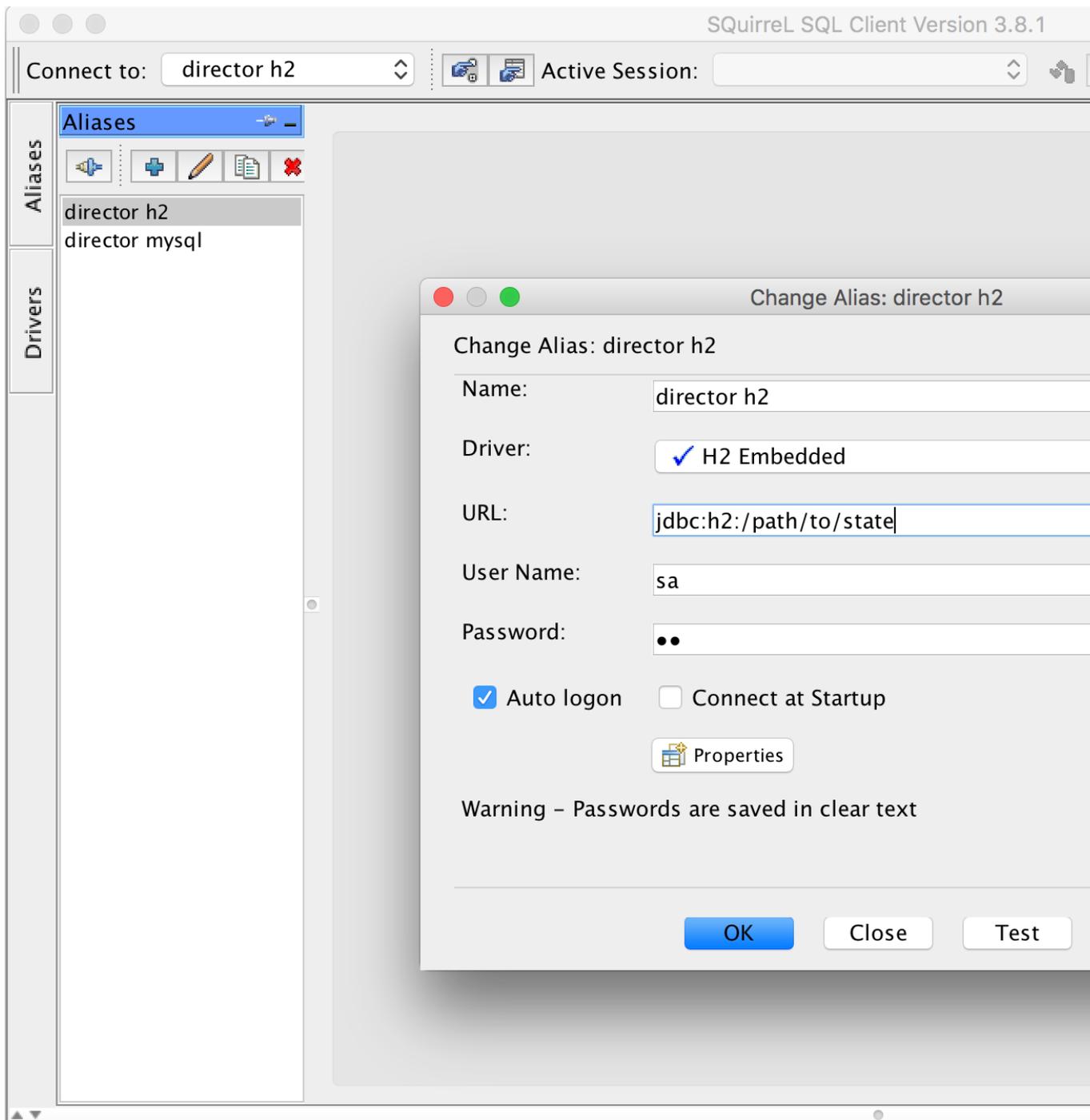
- a. Download H2 from <http://www.h2database.com/html/download.html>.
- b. Edit the **H2 Embedded** driver, adding the downloaded jar file to the **Extra Class Path**.



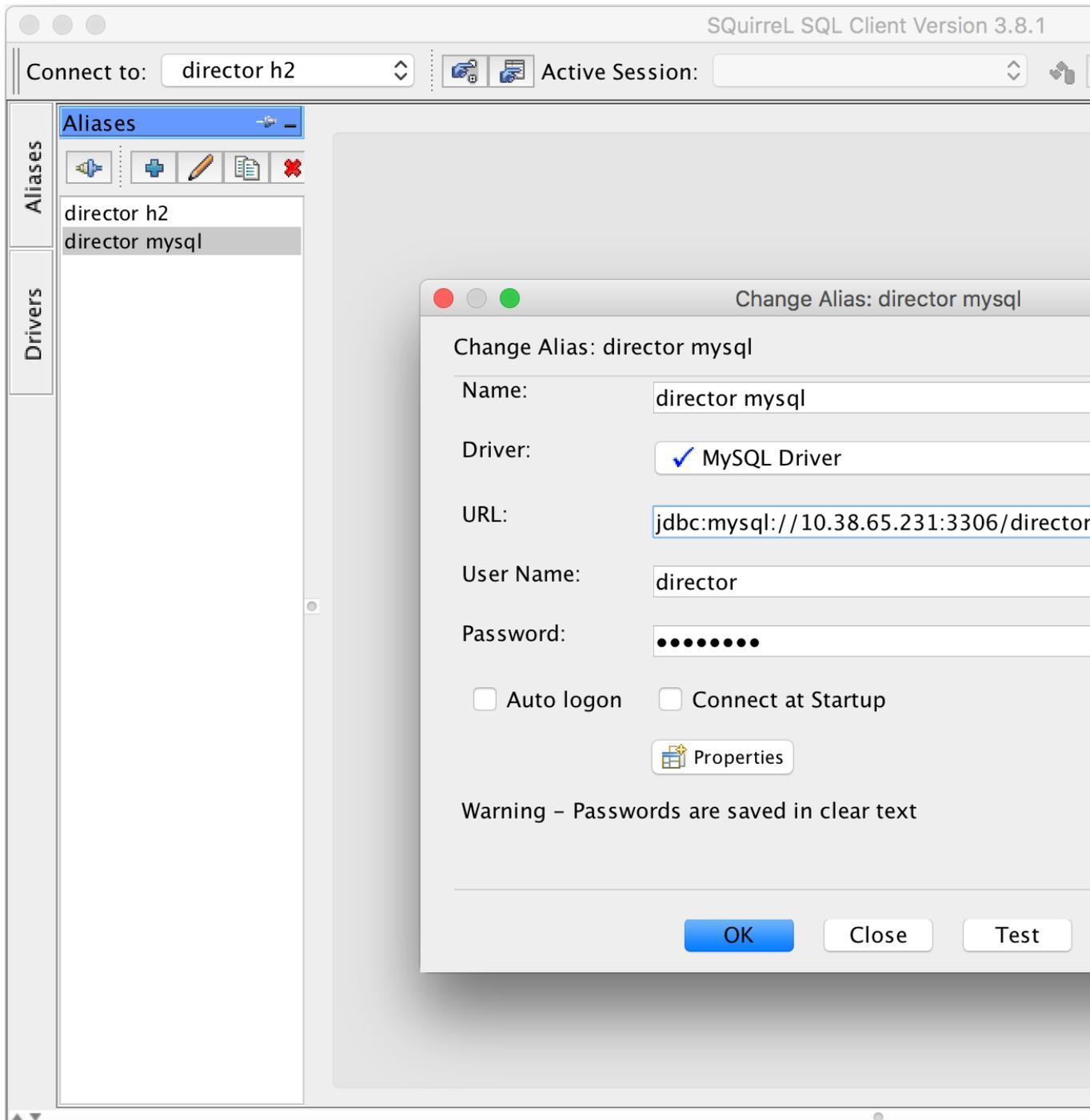
3. Enable the “MySQL Driver” driver in Squirrel SQL.
 - a. Download the MySQL driver from the [MySQL Downloads](#) site.
 - b. Edit the “MySQL Driver” driver, adding the MySQL connector jar to the Extra Class Path.



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.

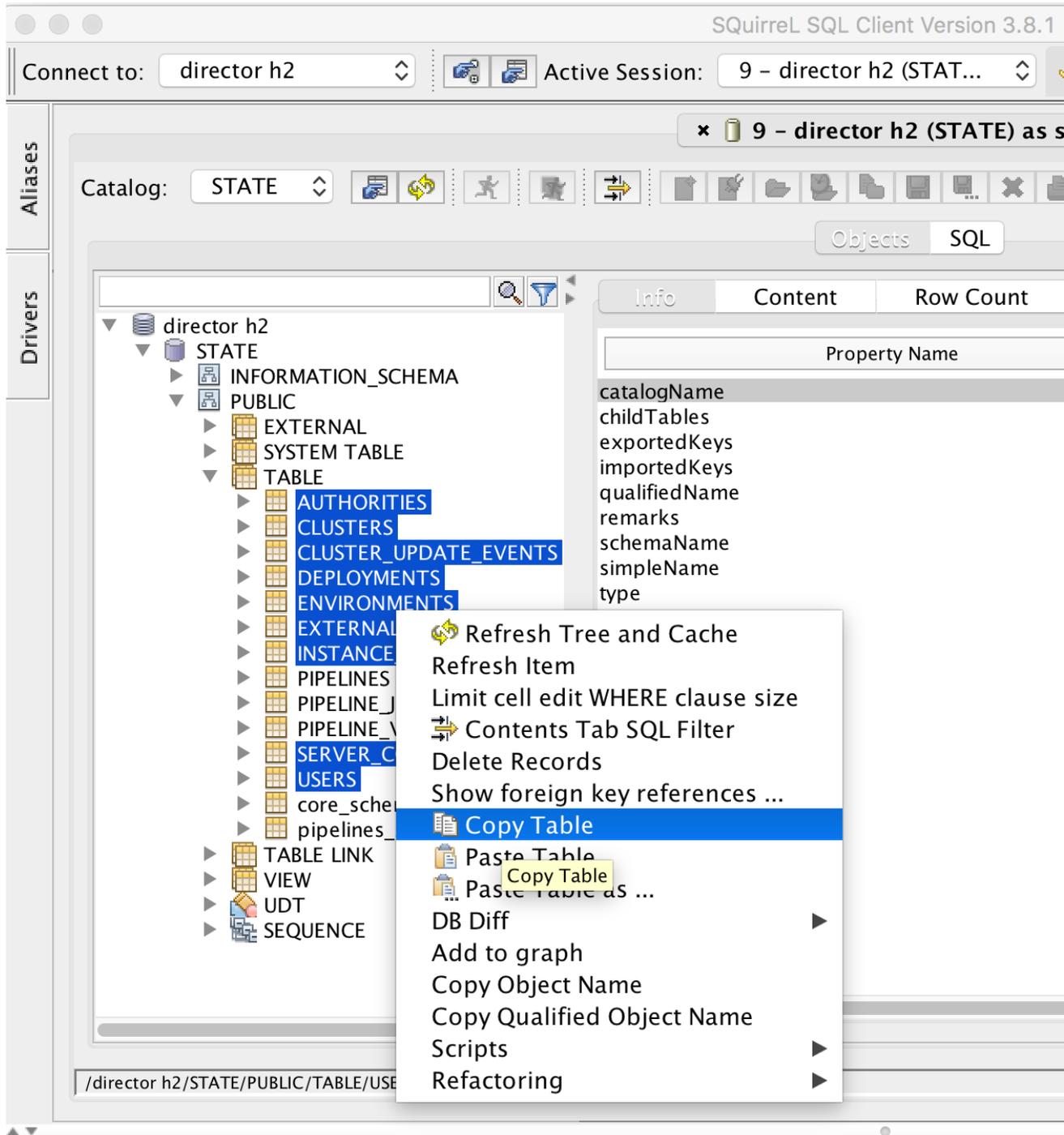
- a. Depending on the Altus Director and MySQL versions you are running, you might need the following step. If you see the error message **invalid default value** after performing the [next step](#), you need to do this. Otherwise, this step is optional.

```
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.
```

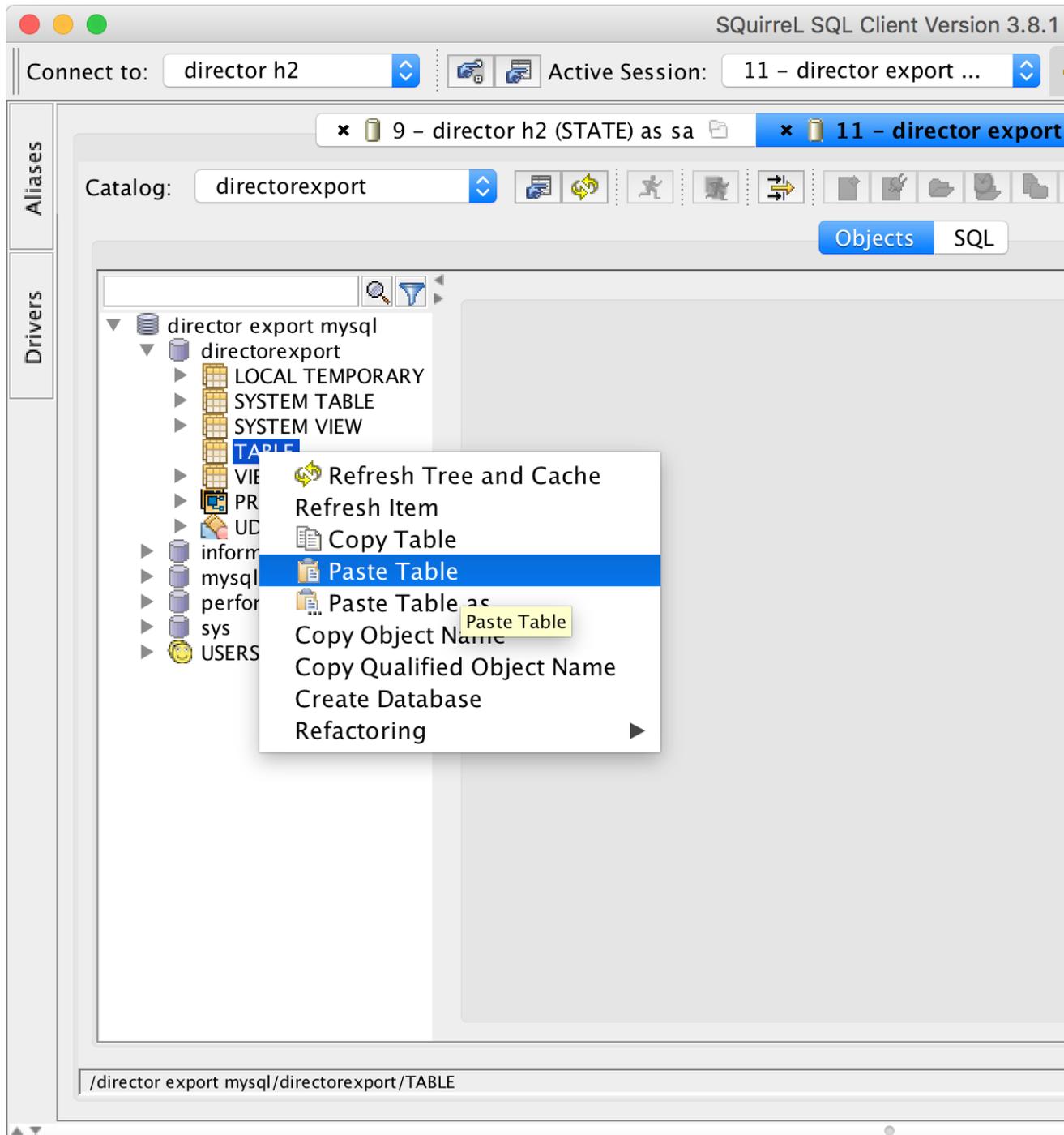
- b. Reconnect to the MySQL alias for this change to apply to your session.

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 Altus Director with MySQL.

1. Configure Altus Director to use MySQL, as described in [Configuring Altus Director Server to use the MySQL Database](#) on page 106.
2. Start Altus Director with MySQL. When Altus Director starts, the database schema will be created.
3. Stop Altus Director to prevent modification of the database during data import.

4. Delete all values from the `AUTHORITIES`, `USERS`, and `SERVER_CONFIGS` tables. Altus 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 Altus 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 Altus Director Server



Note: This section is about the data Altus 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 124.



Note: For information on configuring Altus Director and the MariaDB database to communicate using TLS, see [Enabling TLS with Altus Director](#), and especially the section, [TLS for Altus Director's Database](#) on page 150.

Altus 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 116 to verify that your MariaDB configuration meets the requirements for Altus 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 stop mariadb
```

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 Altus 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/mysql.log
pid-file=/var/run/mysql/mysql.pid
```

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

- For RHEL 6:

```
$ sudo chkconfig mysqld on
```

- For RHEL 7:

```
$ sudo systemctl enable mariadb
```

6. Start the MariaDB server:

- For RHEL 6:

```
$ sudo service mysqld start
```

- For RHEL 7:

```
$ sudo systemctl start mariadb
```

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.

Configuring Storage for Altus Director

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 Altus Director Server

You can create the database on the host where the Altus Director server will run, or on another host that is accessible by the Altus 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. Altus 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 Altus 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 Altus Director configuration settings described below in [Configure Altus 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 -hostname -username -password database > /tmp/database-backup.sql
```

Configuring Altus Director Server to use the MariaDB Database

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



Note: If the Altus 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:

```

Altus Director Database Encryption

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



Note: This section discusses data stored in the Altus 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 3: Server Configuration Properties

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

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

Table 4: 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)

Configuring Storage for Altus Director

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

```
ZGVmYXVsdGRpcmVjdG9yZGVzZWR1a2V5
```



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

Starting with Encryption

Altus Director's default configuration for database encryption encrypts new data stored in the Altus Director database. This default configuration uses triple DES encryption, with a default key, to protect data. In a new installation of Altus 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 Altus 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 Altus Director configuration. While Altus Director will function correctly, keep in mind that there are drawbacks: some data needing protection in the database might 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 Altus 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 Altus Director configuration property `lp.encryption.twoWayCipherConfig` to the Base64-encoded key string before starting Altus 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 Altus Director can still read the file.

Establishing More Secure Encryption for Existing Installations

For an existing installation of Altus Director that uses either no encryption at all (including older releases of Altus 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 Altus 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 Altus Director server to use a transitional cipher.



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

If a transitional cipher is configured, Altus 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 Altus Director releases. It also ensures that all data needing protection becomes encrypted.

To configure a transitional cipher:

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

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

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

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

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

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

```
desede;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 performing ongoing work. If any work is waiting to be resumed by the server on startup (for example, bootstrapping a new cluster), then the server will refuse to change encryption and will stop. If this happens, you must configure the server for its old cipher, start it, and wait for that work to resume and be completed.

Changing from a Transitional Cipher to a Normal Cipher

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

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

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

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

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

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

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

Shrinking an H2 Database



Note: You should not use H2 for production clusters. Cloudera strongly recommends using MySQL or MariaDB for production deployments instead of H2. Use of the H2 database in production environments can result in excessive space consumption for database files and slow database access. Also, unlike managed MySQL and MariaDB databases, H2 files are not backed up regularly, which puts your production deployment of Altus Director at risk of data loss. For instructions on migrating from H2 to MySQL or MariaDB, see the following topics:

- [Using MySQL for Altus Director Server](#) on page 102
- [Using MariaDB for Altus Director Server](#) on page 115

If you use an H2 database for Altus Director's data, the database should not be larger than a few megabytes. The H2 database grows when Altus Director runs over a long period of time because the database is not able to reclaim disk space when entries are deleted. As the database file grows larger, the risk of database corruption increases.

You can reduce the size of the H2 database by exporting and importing all the data using H2's **Script** and **RunScript** commands.

1. Back up the existing H2 database file.
2. Stop Altus Director.
3. Make a backup script using H2's **Script** command:

```
# Make a backup script (backup.zip)
# NOTE: Do not include '.h2.db' when specifying the db file
$ java -cp <h2 jar file> org.h2.tools.Script -url \
"jdbc:h2:/var/lib/cloudera-director-server/state;MV_STORE=false; \
MVCC=true;DB_CLOSE_ON_EXIT=TRUE;AUTO_SERVER=TRUE;TRACE_LEVEL_FILE=4;TRACE_LEVEL_SYSTEM_OUT=0"
\
-user sa -password sa -script backup.zip -options compression zip
```



Note: Both code snippets in these instructions include the default location for the H2 database file, `/var/lib/cloudera-director-server/state.h2.db`. Edit this part of the code snippets if your database file is in a non-default location.



Important: In both code snippets, omit `.h2.db` in the path to the database file. Otherwise H2 will look for a `state.h2.db.h2.db` file and create it if it doesn't exist.

4. Delete the old database.

5. Create a new, smaller database from the script using H2's RunScript command:

```
# Create a new database from the script (backup.zip)
# NOTE: Do not include '.h2.db' when specifying the db file
$ java -cp <h2 jar file> org.h2.tools.RunScript -url \
  "jdbc:h2:/var/lib/cloudera-director-server/state;MV_STORE=false; \
  MVCC=true;DB_CLOSE_ON_EXIT=TRUE;AUTO_SERVER=TRUE;TRACE_LEVEL_FILE=4;TRACE_LEVEL_SYSTEM_OUT=0" \
  -user sa -password sa -script backup.zip -options compression zip
```

6. Start Altus Director.

For more information on H2 databases, see the H2 documentation at [H2 Database Engine](#).

Configuring Storage for Cloudera Manager and CDH

This section explains options for configuring storage for Cloudera Manager and CDH. Options include the following:

- Use of an external database for Cloudera Manager and CDH in place of the embedded H2 database.
- Object storage for use by Altus Director, Cloudera Manager, and CDH clusters:
 - Amazon S3 object storage
 - Microsoft Azure ADLS object storage
- Using AWS EBS volumes for block storage in AWS

Using an External Database for Cloudera Manager and CDH

By default, Altus Director configures Cloudera Manager and CDH services, such as Hive, to use the Cloudera Manager embedded PostgreSQL database. You can use Altus 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 Altus 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 133.

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

- **Altus Director client** - Configure external databases in the `cluster.conf` file and launch Altus Director client by issuing the `bootstrap-remote` command.
- **Altus Director server** - Configure external databases for Altus Director server in one of the following ways:
 - Using the Altus Director web UI
 - Using the Altus Director REST API
 - By editing the `cluster.conf` file and launching the Altus Director server with the `bootstrap-remote` command

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

Defining External Database Servers

Altus 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, Altus 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 Altus Director might require when creating a new server. A template also supports a second table of tag data, which Altus 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 Altus 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 <code>/databaseServers/</code>	Define a new database.	Admin required.
GET <code>/databaseServers/</code>	List all database servers.	
DELETE <code>/databaseServers/{name}</code>	Delete a database server definition.	Admin required.
PUT <code>/databaseServers/{name}</code>	Update a database server definition.	Admin required.
GET <code>/databaseServers/{name}</code>	Get a database server definition.	
GET <code>/databaseServers/{name}/status</code>	Get the status of a database server.	
GET <code>/databaseServers/{name}/template</code>	Get the template from which a database server was defined.	

If a database server template without a host and port is posted to Altus Director, Altus 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 Altus Director, Altus 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	Altus Director is in the process of creating the server.
BOOTSTRAP_FAILED	Altus Director failed to create the server.
READY	The server is available for use.
TERMINATING	Altus Director is in the process of destroying the server.
TERMINATE_FAILED	Altus 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 Altus Director client. Define external database servers in the `databaseServers` section of a configuration file.

Configuring Storage for Cloudera Manager and CDH

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 Altus 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 Altus 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

Altus 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 5.6 and 5.7 RDS instances are supported.
- RDS works through `bootstrap-remote` 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 Altus Director.
- Storage encryption for RDS instances is not supported in Altus Director 2.1.x and lower. Storage encryption is supported in Altus 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 Altus 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 not specify 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 might 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 Altus 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 Altus Director from your platform software repositories.

Key	Description	Example
instanceClass	Instance type for the database server instance	db.m3.medium
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: 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 Altus 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**:

Database Server

Create Database Server Instance | Edit ?

- Create Database Server Instance
- Register Existing Database Server
- Embedded Database (default) ?

Configurations (optional)

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

Create Database Server Instance x

Name *	<input type="text"/>	?
Master username	<input type="text"/>	?
Master user password	<input type="password"/>	?
DB type	MySQL	?
Tags	<input type="button" value="+"/> <input type="text"/>	?
Allocated storage (GB) *	<input type="text"/>	?
Instance class *	<input type="text" value="db.t3.micro"/> ▾	?
DB subnet group name *	<input type="text"/>	?
VPC security group IDs *	<input type="text"/> - <input type="text"/> +	?

➤ Advanced Options

 **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](#), Altus 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:** Altus Director also supports PostgreSQL database servers for Cloudera Manager and CDH, but you must create them outside of Altus 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.

Register Existing Database Server



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

Cancel

OK

Defining a Database Server in AWS Using RDS: API

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

When an external database server template is submitted through POST to the endpoint, and the template lacks a host and port, Altus 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, Altus 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, Altus Director does not terminate the RDS instances. As a result, multiple deployments and clusters can share the same external database servers that Altus Director creates on RDS.

Defining a Database Server in AWS Using RDS: Client Configuration File

The following example defines a server that Altus Director creates using RDS:

```
databaseServers {
  mysqlt1 {
    type: mysql
    user: root
    password: password
    instanceClass: db.m3.medium
    engineVersion: 5.7.40b
    dbSubnetGroupName: default
    vpcSecurityGroupIds: sg-abcd1234
    allocatedStorage: 10
    tags {
      owner: jsmith
    }
  }
}
```

The following example of an external database template uses the new server that Altus 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. Altus 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 Altus 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, Altus 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 Altus Director within the MySQL limit of sixteen characters for usernames.

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

- Reconfigure the database server, and any security measures that apply to it, to allow Altus 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 Altus Director support for external databases.

API

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

Defining External Databases in the Configuration File

External Databases for Cloudera Manager

Define external databases used by Cloudera Manager in the `cloudera-manager` section of a configuration file. The following example defines existing external databases, indicated by the fact that it includes values for the hostnames or IP addresses and the ports.

```
cloudera-manager {
  # ...
  databases {
    CLOUDERA_MANAGER {
      name: scm1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: scmuser
      password: scmpassword
    }
    ACTIVITYMONITOR {
      name: am1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: amuser
      password: ampassword
    }
    REPORTSMANAGER {
      name: rm1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: rmuser
      password: rmpassword
    }
    NAVIGATOR {
      name: nav1
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: navuser
      password: navpassword
    }
    NAVIGATORMETASERVER {
      name: navmetal
      type: mysql
      host: 1.2.3.4
      port: 3306
      user: navmetauser
    }
  }
}
```

```
        password: navmetapassword
    }
}
```

The following example, which does not include hostnames or IP addresses and ports, defines new external databases that Altus Director must create while bootstrapping the deployment.

```
cloudera-manager {
  # ...
  databaseTemplates {
    CLOUDERA_MANAGER {
      name: cmtemplate
      databaseServerName: mysql1
      databaseNamePrefix: scm
      usernamePrefix: cmadmin
    }
    ACTIVITYMONITOR {
      name: cmamtemplate
      databaseServerName: mysql1
      databaseNamePrefix: am
      usernamePrefix: cmamadmin
    }
    REPORTSMANAGER {
      name: cmrmtemplate
      databaseServerName: mysql1
      databaseNamePrefix: rm
      usernamePrefix: cmradmin
    }
    NAVIGATOR {
      name: cmnavtemplate
      databaseServerName: mysql1
      databaseNamePrefix: nav
      user: cmnavadmin
    }
    NAVIGATORMETASERVER {
      name: cmnavmetatemplate
      databaseServerName: mysql1
      databaseNamePrefix: navmeta
      usernamePrefix: cmnavmetaadmin
    }
  }
}
```

Each template must refer to a database server defined elsewhere in the configuration file. The database server template can be for a server that does not yet exist; in that case, Altus 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 Altus 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, Altus 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.

Using EBS Volumes for Cloudera Manager and CDH

Altus 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.

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.

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 on AWS](#) on page 48.



Note: If auto-repair is enabled on a cluster that uses 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 166.

EBS Volume Types

Altus Director supports the EBS volume types gp2, io1, 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.
io1	4 GiB - 16 TiB	Provisioned IOPS SSD (solid state drive) volume. Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads.

EBS volume type	Minimum and Maximum Size	Usage
		 Note: When configuring io1 volumes in an instance template, specify the IOPS value (the number of I/O operations per second to provision for the volume).
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: Altus 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, Altus Director automatically mounts all the instance store volumes that are available. If you *do* specify EBS volumes, Altus 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`.

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.



Note: The configuration settings below do not apply to the root volume, but only to additional EBS volumes. Configuration settings for root volumes depend on the AMIs you have chosen for the instances.

EBS Optimized

EBS Volume Count

EBS Volume Size (GiB)

EBS Volume Type ▼

EBS IOPS Count

Enable EBS Encryption

EBS KMS Key ID

- **EBS Optimized:** Specify whether to enable EBS Optimized I/O. This optimization isn't available with all instance types. Some instance types are EBS optimized by default regardless of this flag. Additional usage charges may apply when using an EBS-optimized instance.
- **EBS Volume Count:** The number of EBS volumes to mount. Altus Director will create and attach these volumes to the provisioned instance. These added volumes will be deleted when the instance is terminated from Altus Director.
- **EBS Volume Size (GiB):** The size of the additional EBS volumes to mount. Specifying a size outside the ranges defined in the table above causes cluster deployment to fail.
- **EBS Volume Type:** The EBS volume type for the additional EBS volumes. Supported volumes are Throughput Optimized HDD (st1), Cold HDD (dc1), General Purpose SSD (gp2), and Provisioned IOPS SSD (io1). All EBS volumes for an instance must be of the same type.
- **EBS IOPS Count:** The number of I/O operations per second (IOPS) to provision for the volume. Only valid and required for Provisioned IOPS (io1) SSD volumes.
- **Enable EBS Encryption:** Whether to enable encryption for the additional EBS volumes. Note that the encryption does not apply to the root volume.
- **EBS Key ID:** The full ARN of the KMS Custom Master Key (CMK) to use when encrypting volumes. If encryption is enabled and this is blank, the default CMK will be used. Note that encryption does not apply to the root volume.

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, scl, gp2 or io1 volume type
# ebsVolumeSizeGiB: 500
# ebsIops: 500           # Number of IOPS, only valid and required for io1 volume
type
#
# 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
#
# EBS Optimized
#
# Specify whether to enable EBS Optimized I/O. This optimization isn't available
# with all instance types. Some instance types are EBS Optimized by default
# regardless of this flag. Additional usage charges may apply when using an
# EBS-optimized instance.
#
# See http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSOptimized.html
#
# ebsOptimized : false
```

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, Altus 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 Altus 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 189.

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.

Creating an Encrypted Root Volume

AWS does not support creating an encrypted root volume as part of the process of spinning up an EC2 instance. To create an encrypted root volume, begin with an AMI that already has an encrypted root volume. Instances launched from such an AMI will also have encrypted root volumes. The AWS account that Altus Director uses must have the necessary permissions to use an AMI with an encrypted root volume.



Note: If you use an AMI with an unencrypted root volume, and configure Altus Director to encrypt the additional EBS volumes, Altus Director will not also encrypt the root volume. If the AMI *does* have an encrypted root volume, it will still be encrypted for the new instance.

Configuring Device Names for EBS Volumes and Instance Store Volumes

When requesting EC2 instances in Altus Director with additional EBS volumes or requesting an instance that contains instance store volumes, Altus Director will automatically assign device names to the volumes. For more information about device names in EC2, see [Device Naming on Linux Instances](#) in the AWS documentation. You can configure the

way the device names are assigned to the volumes. This might be necessary to ensure that the device names used by Altus Director doesn't overlap with any additional volumes associated with an AMI.

By default, AWS creates instance store volumes with device names `/dev/sdb`, `/dev/sdc`, `/dev/sdd`, and so on. You can configure the device name prefix and starting character by adding the following section in `etc/aws-plugin.conf` under the AWS plugin directory.

```
ephemeralDeviceMappings {
  deviceNamePrefix: /dev/sd
  rangeStart: b
}
```

By default, AWS creates EBS volumes with device names `/dev/sdf`, `/dev/sdg`, `/dev/sdh`, and so on. The device name prefix and starting character can be configured by adding the following section in `etc/aws-plugin.conf` under the AWS plugin directory.

```
ebsDeviceMappings {
  deviceNamePrefix: /dev/sd
  rangeStart: f
}
```

Note that Altus Director does not attach both instance store volumes and EBS volumes at the same time. If you specify EBS volumes, instance store volumes will not be attached.

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 might 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 Altus 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 Altus Director

Cluster access to Amazon S3 storage can be configured through Altus Director by launching your cluster with a configuration file and the `bootstrap-remote` CLI command. Altus 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**.

Configuring Storage for Cloudera Manager and CDH

- 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 Altus 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 Altus Director that is configured to use external accounts, you will see the external accounts in the Altus 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 Altus Director web UI. In this case, you must use Cloudera Manager to view all external accounts for the cluster.

- 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)
]
```

- 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  
}
```

Security, Encryption, and High Availability

This section explains how to use security, encryption, and high availability features supported in Altus Director.

Enabling TLS with Altus 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 Altus Director, in order to protect communications among them.



Note: This document assumes you are familiar with TLS and SSL concepts like public and private keys, public key certificates, and certificate authorities. Cloudera recommends that you also be familiar with enabling TLS for Cloudera Manager and CDH. For more information, see [Configuring Cloudera Manager Clusters for TLS/SSL](#) and [Configuring TLS/SSL Encryption for CDH Services](#). For a list of documentation topics on using TLS with Cloudera Manager and CDH, search for **TLS** using the **Categories** feature at the bottom of any page of the [Cloudera Manager or CDH documentation](#), or [click here](#).

Automatic TLS (Auto-TLS)

Altus Director 2.6 and higher can work in concert with Cloudera Manager 5.13 and higher to automatically configure TLS for Cloudera Manager and CDH. This is called auto-TLS. Auto-TLS replaces the work otherwise performed manually to create key pairs and public key certificates, copy them into the correct locations for Cloudera Manager and CDH, and alter Cloudera Manager and CDH component configurations to use them and support access over TLS. Auto-TLS configures TLS as soon as possible, often before relevant services are started, so there is no span of time when services are listening for unencrypted traffic.

Auto-TLS creates a bespoke certificate authority (CA) on the Cloudera Manager server instance. All certificates generated by auto-TLS are signed by this CA. Since the CA does not exist before Cloudera Manager is installed, and because the root certificate for the CA is not itself signed by a pre-existing CA, it is difficult to establish trust for the certificates ahead of time. If you wish to have certificates signed by a pre-existing CA instead of the bespoke CA, then continue to set up TLS manually, and follow the instructions in [Manual TLS](#) to configure Altus Director to communicate with Cloudera Manager over TLS.

Using Auto-TLS

To use auto-TLS, you must either use an Altus Director configuration file or Altus Director's REST API for bootstrapping Cloudera Manager. Auto-TLS is not available through the Altus Director UI.

To enable auto-TLS for a Cloudera Manager deployment using a configuration file, simply include the `tlsEnabled` property in the `cloudera-manager` section, set to `true`.

```
...
cloudera-manager {
    tlsEnabled: true
}
...
```

To enable auto-TLS for a Cloudera Manager deployment using the Altus Director REST API, simply set the `tlsEnabled` property of the deployment template to `true`. You must use at least version 10 of the API. An example JSON snippet is below.

```
...  
  "tlsEnabled": true,  
  ...
```

When auto-TLS is enabled for a Cloudera Manager deployment, every CDH cluster managed by that Cloudera Manager installation is also automatically configured for TLS. If it is necessary for a CDH cluster to not be configured for TLS while Cloudera Manager is, then follow the instructions for [Manual TLS](#) to configure Cloudera Manager instead.

Use of auto-TLS causes the automatic installation of unlimited strength JCE policy files on the Cloudera Manager instance by Altus Director. The policy files eliminate runtime restrictions on the strength of cryptographic algorithms that are included in Java.



Important: If you are in a legal jurisdiction where use of unlimited strength JCE policy files is prohibited, then do not use auto-TLS. Instead, follow the instructions for [Manual TLS](#).

Altus Director automatically retrieves the root certificate of the bespoke CA used in auto-TLS from the Cloudera Manager instance, and establishes trust in it. This creates an opportunity for a man-in-the-middle attack on the SSH connection Altus Director uses for the retrieval. For protection against such an attack, consider configuring [SSH host key retrieval and verification](#).

Retrieving the CA Root Certificate

When a browser or other web client accesses a Cloudera Manager instance set up under auto-TLS, the browser or web client should emit a warning that the server certificate for Cloudera Manager is not trusted. This is because the certificate is signed by the bespoke CA resident on the Cloudera Manager instance, and that CA is not trusted by any of the well-known default CAs installed in the client. It is possible to override or ignore these warnings, but a more secure option is to import the root certificate of the CA into the client.

There are a few ways to retrieve the root certificate of the bespoke CA, so that it can be imported into the trusted certificate store of a client. One easy way is to use the Altus Director REST API, or an API client library, to get the deployment corresponding to the Cloudera Manager instance. The result data includes the root certificate as the `trustedCertificate` field. The value of the field is a single line containing the entire certificate contents; to use the value as a proper encoded certificate, replace each continuous sequence of `\r` and `\n` strings with a single newline.

`http://director_host:7189/api/v10/environments/env_name/deployments/dep_name`

```
{  
  "name": "dep_name",  
  "hostname": "203.0.113.101",  
  "port": 7183,  
  ...  
  "tlsEnabled": true,  
  "trustedCertificate": "-----BEGIN CERTIFICATE-----\r\nMII... \r\n-----END  
CERTIFICATE-----\r\n",  
  ...  
}
```

Once the root certificate is imported for a client, it should be trusted, and no connection warnings should appear.

Advanced Auto-TLS Configuration

Auto-TLS requires no detailed configuration in order to work properly. However, there are a set of TLS configuration properties that can be passed through an Altus Director deployment template to Cloudera Manager. The properties are listed below; each property's key and value is a string.

Table 5: TLS Configuration Properties

Property Name	Purpose	Default Value
<code>trusted_ca_certs</code>	path to a file containing a concatenation of certificates to be trusted by cluster services	none; see below
<code>subject_suffix</code>	suffix for DNs generated for auto-TLS	"ST=CA,C=US"
<code>ca_dn</code>	complete DN for CA root certificate (overrides <code>ca_name</code> and <code>subject_suffix</code>)	none
<code>ca_name</code>	CN for subject DN of CA root certificate, used in concert with <code>subject_suffix</code> ; cannot exceed 64 characters	"SCM Local CA on deployment <i>deployment-name</i> ", truncated if necessary to 64 characters
<code>email_address</code>	email address to include as subject alternative name (SAN) on all certificates	none
<code>ca_key_algo</code>	algorithm used to generate CA key: RSA, DSA, or EC	RSA
<code>ca_key_args</code>	arguments for CA key generation: for RSA and DSA, key length; for EC, curve name	3072
<code>ca_sig_hash_algo</code>	hash algorithm for CA key signature: SHA256, SHA512	SHA256
<code>host_key_algo</code>	algorithm used to generate host keys: RSA, DSA, or EC	RSA
<code>host_key_args</code>	arguments for host key generation: for RSA and DSA, key length; for EC, curve name	3072
<code>host_sig_hash_algo</code>	hash algorithm for host key signatures: SHA256, SHA512	SHA256
<code>ca_expiration</code>	date of CA certificate expiration, in YYMMDD format	about one year from current time
<code>host_expiration</code>	date of host certificate expirations, in YYMMDD format	about one year from current time
<code>key_encryption_algo</code>	encryption used for private keys stored on Cloudera Manager instance: AES128, or AES256	AES256
<code>keytool</code>	path to Java keytool on Cloudera Manager instance	see below

The `trusted_ca_certs` TLS configuration property is useful when cluster services need to connect over TLS to external endpoints, and the server certificates for those endpoints are not ultimately signed by widely acknowledged CAs. The self-signed server or CA certificates for those endpoints, in PEM format, should be included in the file indicated by the property. Since the property value is a path to a file containing the certificates, and not the certificates themselves, measures must be taken to ensure that the file is in place at the indicated path on the Cloudera Manager instance; for example, the file could be included in the instance's base image, or the file could be written through a bootstrap script.

The `host_expiration` and `ca_expiration` TLS configuration properties set the expiration dates for server certificates issued for cluster services and for the Cloudera Manager server, respectively. They default to a date about one year in the future from the time when the certificates are issued. Beyond the expiration dates, the certificates are not accepted by TLS clients. Auto-TLS does not support the renewal of automatically generated certificates, so for long-lived clusters, specify expiration dates further in the future.

If the `keytool` TLS configuration property is passed to Altus Director, then its value is used as is. Otherwise, Altus Director attempts to locate `keytool` on the running Cloudera Manager instance. If Altus Director succeeds in locating `keytool`, it passes the path to `keytool` to Cloudera Manager; otherwise, it passes a default of `keytool`, which works if `keytool` is on the default `PATH`.

To set a TLS configuration property using a configuration file, include its key and desired value in a simple object named `tlsConfigurationProperties` in the `cloudera-manager` section. If no TLS configuration properties need to be set, the `tlsConfigurationProperties` object can be omitted.

```
...
cloudera-manager {
    tlsEnabled: true
    tlsConfigurationProperties {
        ca_key_args: 4096
        host_key_args: 4096
    }
}
...
```

To set a TLS configuration property using the Altus Director REST API, include it in the `tlsConfigurationProperties` map property of the deployment template. An example JSON snippet is below. If no TLS configuration properties need to be set, the `tlsConfigurationProperties` map property can be omitted or left empty.

```
...
"tlsEnabled": true,
"tlsConfigurationProperties": {
    "ca_key_args": "4096",
    "host_key_args": "4096"
},
...
```

Altus Director does not attempt to validate TLS configuration properties, so any unknown or invalid properties are passed through as-is to Cloudera Manager. Check Altus Director and Cloudera Manager logs if the configuration properties appear to cause problems or have no effect.

Name Length Restrictions

Some standard naming restrictions are relevant when configuring TLS.

- The fully-qualified domain name (FQDN) of a Linux instance can be up to 255 characters in length.
- The value of the common name (CN) attribute in a distinguished name (DN) cannot exceed 64 characters.

Because auto-TLS normally uses the FQDN of an instance as the value of the CN attribute in that instance's certificate's subject DN, it is possible for certificate generation to fail if that FQDN is too long. When the FQDN exceeds 64 characters, a certificate with the FQDN as the value for the CN attribute cannot be generated.

This is of particular importance when using Google Compute Engine, which includes an instance's Altus Director virtual instance ID, a 36-character UUID, in the instance's hostname. Google Compute Engine also uses the project name as part of the domain name, consuming more characters from the allowed total of 64.

Therefore, when using auto-TLS, you might need to specify a short instance name prefix for Altus Director instance templates in order to stay below the 64-character limit. In the Altus Director UI, the instance name prefix is specified under the Advanced Options when creating or editing an instance template. In an Altus Director configuration file, the

instance name prefix can be specified in the "provider" section when using either AWS, Google Cloud Platform, or Microsoft Azure as a cloud provider.

```
provider {
  instanceNamePrefix: abc
  ...
}
```

Static Private IP Addresses

In order to be compatible with Altus Director, the server certificate for the Cloudera Manager instance must include its private IP address as a subject alternative name (SAN). If the instance changes its private IP address, perhaps by being stopped and started again, then Altus Director will not be able to communicate with Cloudera Manager over TLS due to the mismatch.

There is currently no mechanism in auto-TLS to accommodate a change in the private IP address of the Cloudera Manager instance. Therefore, it is essential that the private IP address for the Cloudera Manager instance be assigned until the instance is terminated, and not released when it is stopped.

This is the normal behavior for instances under AWS and Google Cloud Platform. However, the default behavior for Microsoft Azure is to use *dynamic* private IP address allocation, which can cause the private IP address to change when the VM is stopped and then started again. In contrast, a *static* private IP address allocation remains with a VM until it is terminated. To change the private IP address allocation for an Azure VM from dynamic to static, use the [Azure portal](#) or [Powershell](#). For more information about configuring Altus Director on Azure to use static IP addresses, see the first note in [Notes on Pausing a Cluster](#) on the page [Pausing Altus Director Instances](#).

Coping with Untrusted Certificates Created by Auto-TLS

Server certificates signed by the bespoke CA are not trusted by web browsers or other clients, leading them to either produce warnings or refuse to complete TLS handshakes. There are several strategies for coping with the certificates.

First, clients can be configured to trust the root certificate of the CA by importing that certificate into the client's trusted certificate store. Once this is done, the client trusts any server certificates signed by the CA, just as it would for those signed by widely recognized CAs. However, this process must be performed for each new CA that is created by auto-TLS, and it usually requires manual steps.

A more user-friendly solution is to place a [reverse proxy](#) in front of Cloudera Manager and the cluster services. A reverse proxy can be configured to listen over TLS, using a server certificate signed by a recognized CA so that clients need not import any new trusted certificates. The server name for the proxy can be in your organization's domain and managed by your organization's CA, whether internal or third-party. In response to requests, the reverse proxy then can route new requests, again over TLS, to Cloudera Manager and cluster services.

For example, here is a configuration snippet for the [nginx](#) HTTP server and proxy that routes incoming TLS requests to the HDFS namenode web UI, also serving over TLS. In the example, `hadoopproxy` is the hostname of the instance hosting `nginx`, and `nn.cloudprovider.internal` refers to the instance hosting the HDFS namenode. A client request to `https://hadoopproxy/` connects to the HDFS namenode web UI, even if the client cannot reach the HDFS namenode instance directly.

```
server {
  listen 443 ssl;
  server_name hadoopproxy;

  # public key certificate for nginx, signed by recognized CA
  ssl_certificate /etc/nginx/cert.pem;
  # private key for nginx
  ssl_certificate_key /etc/nginx/key.pem;

  location / {
    proxy_pass https://nn.cloudprovider.internal:50470/;
    # root certificate for bespoke CA on Cloudera Manager instance
    proxy_ssl_trusted_certificate /etc/nginx/cmca.crt;
    proxy_ssl_verify on;
    proxy_ssl_verify_depth 2;
  }
}
```

```
    proxy_ssl_session_reuse on;
  }
}
```

Cloud provider load balancing services can also work as reverse proxies. For example, an application load balancer created through AWS Elastic Load Balancing (ELB) can be configured as a reverse proxy, listening for requests over TLS using a server certificate signed by a recognized CA and forwarding them over TLS to Cloudera Manager or cluster service web UIs. Check the documentation for your cloud provider for the available options.

Manual TLS

You might wish to set up TLS manually for Cloudera Manager and CDH instead of through the auto-TLS capability. One common reason for doing so is in order to use certificates signed by an existing CA, as opposed to the bespoke CA set up under auto-TLS. Altus Director supports this workflow by allowing you to enable TLS, via the Altus Director API, on a bootstrapped deployment.

The process of configuring TLS manually for Cloudera Manager and CDH is documented in [Configuring Cloudera Manager Clusters for TLS/SSL](#), and the same process should be followed for installations bootstrapped by Altus Director. After the initial part of the process, **Level 0**, Altus Director must be updated to use TLS for communication. The remaining parts of the process should still be performed.

Configuring Level 0 for Cloudera Manager

Start by bootstrapping a deployment and cluster normally, without TLS enabled. Once the deployment and cluster are READY in Altus Director, perform the usual manual steps for configuring TLS in Cloudera Manager, with a slight modification explained below. See [Level 0: Basic TLS/SSL Configuration](#) for more information. The only steps that must be performed before updating Altus Director are those for Level 0, which reconfigure the Cloudera Manager server to use TLS. For complete Cloudera Manager TLS configuration, after updating Altus Director, be sure to continue the steps after Level 0 as usual for securing Cloudera Manager and CDH manually.

In order to be compatible with Altus Director, the server certificate for Cloudera Manager must include a *subject alternative name* (SAN) with the private IP address of the Cloudera Manager instance. The stock instructions for configuring Level 0 do not include a SAN, but `keytool` allows specifying one using the `-ext` option. The example below illustrates how to use the `-ext` option with an IP address and a hostname; replace the sample IP address with the correct one when used. Because TLS clients might ignore the hostname in the common name (CN) attribute of the certificate's subject DN in lieu of SANs, you should include the hostname as a SAN as well.

```
$ sudo keytool -genkeypair -alias $(hostname -f)-server -keyalg RSA \
-keystore /opt/cloudera/security/pki/$(hostname -f)-server.jks \
-keysize 4096 -dname "CN=$(hostname -f),O=cloudera.com,ST=CA,C=US" \
-ext "san=ip:203.0.113.101,dns:$(hostname -f)" \
-storepass cloudera -keypass cloudera
```

During Level 0 configuration, you have the opportunity to have the public key certificate for the server signed by a CA. If desired, you can proceed with a self-signed certificate. To generate a self-signed certificate from the server key pair, use `keytool`. The self-signed certificate must be imported into the Java truststore (the file `jssecacerts` created in the Level 0 documentation), just as the root certificate for an internal CA would need to be. Be sure to perform this extra step after generating a self-signed certificate, instead of generating a certificate signing request (CSR).

```
$ sudo keytool -exportcert -rfc \
-keystore /opt/cloudera/security/pki/$(hostname -f)-server.jks \
-alias $(hostname -f)-server \
-file /opt/cloudera/security/pki/$(hostname -f)-server.crt \
-storepass cloudera -keypass cloudera
```

A self-signed certificate is untrusted by browser and other web clients, so there is often no reason to use one instead of taking advantage of auto-TLS.

Updating Altus Director

Any time after Level 0 configuration is complete, it is possible to update the deployment information in Altus Director to enable TLS communication. Two pieces of information are required:

1. The encrypted port number for Cloudera Manager. This is usually 7183.
2. Either the self-signed certificate for the Cloudera Manager server or, if using an internal CA, either the signed public key certificate for the server or the root certificate for the internal CA. No matter which certificate you choose, it is called the trusted certificate for the deployment.

Retrieve the deployment template for Cloudera Manager from Altus Director's REST API. You must use version 10 or higher of the Altus Director API. API access is available through the API console at `http://director_host:7189/api-console/`, or through an Altus Director API client library, or directly using your preferred REST client framework.

```
http://director_host:7189/api/v10/environments/env_name/deployments/dep_name/template
```

Edit the returned JSON to add or change the following fields in the top-level object:

- `tlsEnabled`, with a value of `true`
- `port`, with the port number as its numeric value, even if the default port of 7183 is to be used
- `trustedCertificate`, with a string value matching the contents of the trusted certificate. Because the certificate is multiple lines, remove all newline characters except for one at the start of the key and one at the end of the key (as in the example below).

Here is an example of the new fields, with most of the contents of the trusted certificate omitted for brevity. The order and placement of the three fields isn't important, but they must be top-level fields in the JSON object:

```
"name": deployment_template_name # this 'name' is already in your JSON template
"tlsEnabled": true,
"port": 7183,
"trustedCertificate": "-----BEGIN CERTIFICATE-----\nMII...\n-----END CERTIFICATE-----"
```

Finally, update the deployment by issuing an HTTP PUT to the deployment update endpoint. Include the entire modified deployment template in the request body, not just the new fields.

```
http://director_host:7189/api/v10/environments/env_name/deployments/dep_name
```

During the deployment update process, Altus Director attempts to connect to Cloudera Manager over TLS, using the port and trusted certificate supplied in the update. The update will fail if Altus Director cannot successfully connect.

Completing Manual TLS

After successful deployment update, Altus Director will communicate with Cloudera Manager exclusively over TLS. Be sure to return to the documented process of configuring TLS for other components of Cloudera Manager, such as its agents, to complete the work of securing Cloudera Manager and CDH clusters.

Disabling Manual TLS

If it becomes necessary to disable TLS for Cloudera Manager, it is possible to update Altus Director so that it communicates over unencrypted HTTP to Cloudera Manager. Performing such an update is only necessary if TLS is disabled for the Cloudera Manager API and web interface; if TLS is only disabled, say, for Cloudera Manager agent communication, then Altus Director does not need to be updated. In other words, Altus Director should be updated only if Level 0 is unconfigured for Cloudera Manager.

Also, if TLS is disabled only temporarily for Cloudera Manager, and then re-enabled soon afterward, then Altus Director will re-establish communication on its own. While TLS is disabled for Cloudera Manager, Altus Director will be unable to communicate with it. Often this is not a problem. However, health reports, refresh processes, and usage-based billing will not function while Altus Director is out of contact.

Disabling TLS for Cloudera Manager should not be necessary when auto-TLS is in use. These instructions assume that TLS was configured manually for Cloudera Manager.

Begin by retrieving the deployment template for Cloudera Manager from Altus Director's REST API. You must use version 10 or higher of the Altus Director API. API access is available through the API console at `http://director_host:7189/api-console/`, or through an Altus Director API client library, or directly using your preferred REST client framework.

```
http://director_host:7189/api/v10/environments/env_name/deployments/dep_name/template
```

Edit the returned JSON to add or change the following fields in the top-level object:

- `tlsEnabled`, with a value of `false`
- `port`, with the port number as its numeric value, even if the default port of 7180 is to be used
- `trustedCertificate`, which should be omitted completely from the template

Here is an example of the updated fields:

```
"tlsEnabled": false,  
"port": 7180
```

Finally, update the deployment by issuing an HTTP PUT to the deployment update endpoint. Include the entire modified deployment template in the request body, not just the updated fields.

```
http://director_host:7189/api/v10/environments/env_name/deployments/dep_name
```

During the deployment update process, Altus Director attempts to connect to Cloudera Manager over an unencrypted connection, using the port supplied in the update. The update will fail if Altus Director cannot successfully connect.

After successful deployment update, Altus Director will communicate with Cloudera Manager without TLS. If or when TLS is enabled once again for Cloudera Manager, repeat the procedure above for updating Altus Director to communicate over TLS.

Working with an Existing Cloudera Manager

Altus Director allows you to bootstrap a cluster using a deployment of Cloudera Manager that already exists, and was not itself created by Altus Director. When using an existing Cloudera Manager, auto-TLS is not available. However, Altus Director can still work with an existing Cloudera Manager installation over TLS.

If the existing Cloudera Manager installation is already configured for TLS, then the Altus Director configuration file should indicate that in the `cloudera-manager` section by:

- setting the `tlsEnabled` property to `true`
- optionally setting the `port` property to the encrypted port (or else the default of 7183 is used)
- setting the `trustedCertificate` property to a string value matching the contents of the trusted certificate; because the certificate is multiple lines, replace each newline in it with a two-character `\n` string

If the existing Cloudera Manager installation is not configured for TLS when the cluster is bootstrapped, but TLS is then manually enabled later, then you should update the deployment information in Altus Director. See [Updating Altus Director](#) above for instructions on how to do so; the procedure is identical to that for a Cloudera Manager installation that was originally bootstrapped by Altus Director. As noted above, the server certificate for Cloudera Manager must include a *subject alternative name* (SAN) with the private IP address of the Cloudera Manager instance in order to be compatible with Altus Director.

TLS for Altus Director Itself

Altus Director itself can be configured for TLS. When this is done, the Altus Director server, including its API endpoints, require TLS for access. Configuring Altus Director itself for TLS is an independent activity from configuring Cloudera Manager and CDH clusters for TLS, although a comprehensive security plan might mandate them both. Altus Director can be configured for TLS either before or after any Cloudera Manager deployments.

Configuring Altus Director for TLS

Altus Director is configured for TLS by setting its own configuration properties in its `application.properties` file. See [Setting Altus Director Properties](#) for information on where the file is located and details on how to set properties.

Property	Description
<code>server.ssl.key-store</code>	Path to keystore holding server's private key and public key certificate.
<code>server.ssl.key-store-password</code>	Password for keystore.
<code>server.ssl.key-store-type</code>	Type of keystore. Optional.
<code>server.ssl.key-store-alias</code>	Alias of private key in keystore, for JKS type keystores. The default is to expect only a single alias in the keystore. Optional.
<code>server.ssl.key-password</code>	Password for private key, if different from password for keystore. Optional.

Additional properties are supported. Consult the [Spring Boot documentation on configuring SSL](#) for details.

After setting TLS configuration properties, restart Altus Director for it to begin requiring TLS. Director continues to listen on the same port as before (default 7189) unless configured for a different port.

Once Altus Director is configured for TLS, access its web interface using https URLs. If the server certificate is not signed by a certificate authority recognized by the client (for example, the web browser), then take the appropriate steps to either add an exception for the server or add the server certificate, or one from a responsible certificate authority, to the client's truststore.

Generating a Test Keystore for Altus Director

To generate a simple test keystore for Altus Director, use the `keytool` command. An example command is below.

```
$ keytool -genkeypair -alias director -keyalg RSA \
  -keystore director.jks \
  -keysize 4096 -dname "CN=$(hostname -f),O=cloudera.com,ST=CA,C=US" \
  -storepass cloudera -keypass cloudera
```

Use a valid hostname for the CN (common name) component of the subject DN (distinguished name) for the public key certificate. Doing so allows clients to perform hostname verification.

Then, configure Altus Director with this keystore to enable TLS. Note that since the public key certificate in the keystore is self-signed, clients such as web browsers will issue warnings that the certificate is not trusted.

```
server.ssl.key-store: /path/to/director.jks
server.ssl.key-store-password: cloudera
```

To extract the self-signed certificate from the test keystore, use `keytool` again, as in the following command. The resulting certificate file can be imported into a truststore using the `keytool -importcert` command.

```
$ keytool -exportcert -rfc \
  -keystore director.jks -alias director \
  -file director.crt -storepass cloudera -keypass cloudera
```

Avoid using a self-signed certificate for Altus Director in a production environment. Instead, use a certificate signed by a recognized certificate authority.

Using the Altus Director Client under TLS

When the Altus Director server is configured for TLS, remote commands issued to the Altus Director client, such as `bootstrap-remote` and `terminate-remote`, must indicate that TLS is enabled on the server. Do so by passing the following configuration properties to the client:

- `lp.remote.tlsEnabled`, set to `true`
- `lp.remote.hostAndPort`, set to the hostname used in the CN component of the subject DN of the server certificate, followed by a colon and the server port

Here is an example bootstrap-remote command that works with an Altus Director server that has TLS enabled.

```
$ cloudera-director bootstrap-remote myconfig.conf \  
--lp.remote.hostAndPort=$(hostname -f):7189 \  
--lp.remote.username=admin --lp.remote.password=admin \  
--lp.remote.tlsEnabled=true
```

Trusting the Altus Director Server Certificate

The Altus Director client relies on the default Java truststore to locate trusted certificates. If the server is using a server certificate that is not signed by a certificate authority whose root certificate is in the truststore, then the client will not be able to connect to the server.

To add a server certificate or certificate authority root certificate to a Java truststore, use a keytool command such as the one below, which assumes that the certificate is being added to a copy of the standard cacerts truststore file.

```
$ cp $JAVA_HOME/jre/lib/security/cacerts directorclients.jks  
$ keytool -importcert -alias director -file director.crt \  
-keystore directorclients.jks -storepass changeit -noprompt
```

To use a custom truststore with the Altus Director client, set the appropriate Java SSL system properties using the `DIRECTOR_CLIENT_JAVA_OPTS` environment variable. At least the "javax.net.ssl.trustStore" property must be set. The contents of the environment variable are passed to the Altus Director client JVM when using the `cloudera-director` run script.

```
$ export DIRECTOR_CLIENT_JAVA_OPTS=\  
"-Djavax.net.ssl.trustStore=/path/to/directorclients.jks \  
-Djavax.net.ssl.trustStorePassword=changeit"  
$ cloudera-director bootstrap-remote ...
```

Disabling Hostname Verification



Note: Disabling hostname verification introduces additional risk when using TLS, and should therefore not be used unless necessary. Proceed at your own risk.

By default, the Altus Director client performs normal hostname verification based on the CN component of the subject DN. Hostname verification can be disabled by passing the `lp.remote.hostnameVerificationEnabled` configuration property to the client, set to `false` (the property's default value is `true`).

This is helpful, for example, when connecting to an Altus Director instance whose certificate specifies a hostname that is private to your cloud provider, but a different, public hostname must be used. (A superior resolution to this problem is to include the public hostname as a subject alternative name in Altus Director's server certificate.)

```
$ cloudera-director bootstrap-remote myconfig.conf \  
--lp.remote.hostAndPort=public-hostname.company.com:7189 \  
--lp.remote.username=admin --lp.remote.password=admin \  
--lp.remote.tlsEnabled=true \  
--lp.remote.hostnameVerificationEnabled=false
```

TLS for Altus Director's Database

By default, when the Altus Director server is configured to use a MySQL or Maria DB database for its own data, it communicates with the hosting MySQL or Maria DB server over an unencrypted connection. You can configure the Altus Director server to communicate with the MySQL or Maria DB server over TLS.



Note: This section is about the data Altus Director server stores for its own use, and not for databases used by Cloudera Manager or cluster services.



Note: For information on configuring Altus Director to use a MySQL database, see [Using MySQL for Altus Director Server](#). For information on configuring Altus Director to use a MariaDB database, see [Using MariaDB for Altus Director Server](#).

Setting up TLS for communication with MySQL or Maria DB adds a few steps to the general procedures described in [Using MySQL for Altus Director Server](#) and [Using MariaDB for Altus Director Server](#). As part of configuring the MySQL server, include the configuration options needed for TLS support. Consult MySQL documentation for complete information, depending on the version of MySQL in use. Some example options:

```
[mysqld]
ssl-ca=ca.pem
ssl-cert=server-cert.pem
ssl-key=server-key.pem
```

In this example:

- **ca.pem** is a file containing the certificate for the certificate authority (CA) that has signed the server certificate
- **server-cert.pem** is a file containing the public key certificate for the server
- **server-key.pem** is a file containing the private key for the server

Configure the set of `lp.database` configuration properties for the Altus Director server as usual to point to the MySQL or Maria DB database server and the Altus Director database created within it. Be sure to explicitly supply values for `lp.database.host`, `lp.database.port`, and `lp.database.name`, even if their defaults apply.

Then, set the value of the `lp.database.url` configuration property to the following (all on one line):

```
jdbc:mysql://${lp.database.host}:${lp.database.port}/${lp.database.name}?verifyServerCertificate=true&useSSL=true&requireSSL=true
```

Finally, import the CA certificate to the Java system truststore. This allows Altus Director to trust the server certificate used by the MySQL server. Use a unique, appropriate alias for the certificate in the truststore. Do this before starting Altus Director.

```
sudo keytool -importcert -keystore $JAVA_HOME/lib/security/cacerts \
-file ca.pem -alias mysqlca
```

As an alternative to importing the certificate into the Java system truststore, you can import the certificate into a separate truststore that is passed to Altus Director via the `javax.net.ssl.trustStore` system property. If you do this, be sure the separate truststore contains all other necessary certificates for Altus Director to function, such as those providing the basis of trust for communicating with your chosen cloud provider, and either Altus Director's own server certificate or a CA certificate in its chain.

When the Altus Director server is started, it will communicate with the MySQL or Maria DB server over TLS. If TLS communications cannot be established, Altus Director will exit.

Altus Director might fail to start and report the error "Path does not chain with any of the trust anchors" in its log. This might occur, depending on the version of Java in use and characteristics of the CA root certificate. If this error appears, then import the server certificate for the MySQL or Maria DB server itself into the truststore and restart the Altus Director server.

Using an AWS RDS MySQL Server

You can use a MySQL server launched from AWS RDS to host Altus Director's database. RDS automatically configures each MySQL database instance to support TLS connections, generating an appropriate server certificate for the instance during launch time. The server certificate is signed by a CA maintained by RDS itself.

You must import a certificate from RDS into the Java truststore for Altus Director to be able to communicate with the RDS instance over TLS. RDS maintains a root CA that governs most of its regions. RDS also maintains intermediate CAs for each region. Either the root CA certificate or the intermediate CA certificate for the region where the MySQL server resides should work to establish trust from Altus Director to the RDS instance over TLS.

Download the RDS root CA certificate from the link available at [SSL Support for MySQL DB Instances](#) in the AWS documentation. Download the RDS intermediate CA certificate for the region where the MySQL server resides from the appropriate link available at [Intermediate Certificates](#) in the AWS documentation.

If using the root CA certificate results in the error "Path does not chain with any of the trust anchors," then switch to using the intermediate CA certificate.

Using Custom DNS with Auto-TLS on AWS

When you use Altus Director to deploy Cloudera Manager and launch CDH clusters in AWS, Amazon assigns private IP addresses and generates private DNS names for the EC2 instances. Altus Director uses these to communicate with the EC2 instances in the cluster. Depending on the needs and policies of your organization, you can configure the following custom DNS settings:

- Custom private DNS names for your EC2 instances. Use of custom private DNS names requires also configuring a custom DNS server (or Amazon Route 53).
- A custom DNS server, instead of the Amazon-provided DNS server on the VPC.

Altus Director 2.6 and above supports auto-TLS, as described in [Enabling TLS with Altus Director](#) on page 141, including auto-TLS with custom configurations for hostnames and DNS servers. The following sections describe how to use custom DNS settings with auto-TLS.



Note: A good starting point for learning about using DNS in AWS is [Using DNS with your VPC](#) in the AWS documentation.

How Altus Director and Cloudera Manager Communicate with Cluster Instances

When working with the AWS DNS settings for EC2 instances created and managed by Altus Director, it is useful to understand how Altus Director and Cloudera Manager access the instances in a cluster on AWS. Whether or not you are using auto-TLS affects the way Altus Director configures Cloudera Manager to communicate among the instances.

How Cloudera Manager Communicates with Cluster Instances

- When auto-TLS is not used, Cloudera Manager accesses cluster instances through their private IP address.
- When auto-TLS is used, Cloudera Manager uses private hostnames, as well as IP addresses, to access the instances. Specifically, Cloudera Manager uses private hostnames to configure the Cloudera Manager agents on the EC2 instances. This is because TLS certificates will be generated using the hostname, and then used for secure communications between Cloudera Manager and the cluster.

How Altus Director Communicates with Cluster Instances

- Regardless of whether or not auto-TLS is used, Altus Director uses the IP addresses and hostnames provided by the AWS metadata service to connect via SSH into the cluster instances. When custom hostnames have been configured on the instances, the hostname will not match the name provided by the AWS metadata service.
- In a cross-region, cross-VPC, or cross-cloud-provider configuration, Altus Director will use the private IP address if it is available, and will fall back on the private hostname, public IP address or hostname, in that order, if it is not. For more information on these configurations, see [Running Altus Director and Cloudera Manager in Different Regions or Clouds](#) on page 99

Using Custom DNS Names and Auto-TLS

When users configure their own custom hostnames, the AWS metadata service does not update its record of cluster hostnames, so the hostname on the instance level is different from the hostname on the cloud provider level.



Important: Because custom hostnames are set on the EC2 instance, not in the AWS metadata, you must use Altus Director 2.7.1 or higher if you use auto-TLS with custom hostnames. Altus Director 2.7.1 uses the hostname retrieved from the instance, while earlier versions of Altus Director used the hostname retrieved from the cloud provider. This means that if your instances have custom hostnames, Altus Director 2.7.1 and higher will use them, but earlier Altus Director versions will try to use the hostname from the AWS metadata, causing cluster installation to fail.

There are two parts to configuring a custom DNS name in AWS:

- Configuring a custom domain name
- Configuring a custom hostname

Configuring a Custom Domain Name in AWS

The domain name is the second part of a DNS name. In the example `ip-10-1-2-3.mycompany.com`, the domain name is `.mycompany.com`. To specify a value for the domain name to be used in the VPC, follow these steps:

1. Create a new DHCP options set.
2. Set the domain name in the DHCP options set for the VPC.
3. Set the IP address of the DNS server in the **Domain name servers** field of the DHCP option set for the VPC.
4. Set `enableDnsSupport` to **false** so that the Amazon-provided DNS server in the VPC is not enabled.
5. Set up your own DNS server (or Amazon Route 53).
6. Add a record for your domain name to your custom DNS server (or Amazon Route 53).

You can use the Amazon-provided hostname with your custom domain name, for example, `ip-private-ip-address.mycompany.com`.

Configuring a Custom Hostname in AWS

The hostname is the first part of a DNS name. In the example `ip-10-1-2-3.mycompany.com`, the hostname is `ip-10-1-2-3`.

1. Set up the hostname. There are many ways to do this, for example, through your AMI, user data, orchestration framework (such as Chef or Ansible), or bootstrap scripts.
2. Add a record for your hostname to your custom DNS (or Route 53).

Using a Custom DNS Server and Auto-TLS

You can configure a custom DNS server by entering its IP address in the **DHCP options set** for the VPC. If you use your own DNS server, ensure that the server can resolve hostnames in the Amazon standard format, `ip-x-x-x-x`, or change the name on the host to a name that the DNS server can resolve. You can either pre-populate your DNS server with the expected names, or use a tool like `nsupdate` to register the hosts in the DNS server as they come up.

To specify a custom DNS server, follow these steps::

1. In the AWS Admin Console, choose **VPC**.
2. Disable the Amazon-provided DNS server in the VPC by setting `enableDnsSupport` to **false**.
 - a. In the VPC dashboard, select the VPC you are using.
 - b. In the **Actions** dropdown, click **Edit DNS Resolution**.
 - c. Set the value to **No**.
3. In the left hand pane, click **DHCP Options sets**.
4. Click **Create DHCP options set**.
5. Enter the IP address of your domain name server in the **Domain Name Servers** field.
6. Optionally, configure other fields in the DHCP options set and click **Create DHCP options set**.
7. In the VPC settings, specify the new DHCP options set for your VPC.
8. Configure your DNS server to accept updates, if possible. If you do not want your DNS server to accept updates (for example, because of your organization's security policies), configure the DNS server with hostnames before

launching the cluster. Work with your infrastructure team to ensure that the hostnames you will use are added to your DNS server.

Whether your domain names use the auto-generated format used by AWS or you create a custom hostname in a format unique to your instances, you need to ensure that your custom DNS server can resolve the cluster hostnames. There are a number of ways to do this: by using a script, by adapting the [scripts on the Cloudera GitHub site](#) that were created for Microsoft Azure, or by manual configuration, or by using an orchestration framework, such as Chef or Ansible.

Using a Custom DNS Server on Altus Director 2.6 or 2.7.0

With Altus Director 2.6 or 2.7.0, you can use your own DNS server as long as your private hostnames use exactly the same format as the Amazon-generated DNS names use, so that there is no difference in DNS names retrieved from the AWS metadata and the DNS names stored in your DNS server. You must register these DNS names in your DNS server.

- The format of an AWS-generated private hostname is `ip-` followed by digits derived from the private IP address of the instance, with dashes in place of the periods, for example:

```
ip-10-1-2-3
```

- The domain name part of an AWS-generated private DNS name depends on the `enableDnsHostnames` and `enableDnsSupport` settings in the VPC.
 - If either of those are set to false, the domain name will be whatever is specified in the **DHCP options set**. The default domain name value in the **DHCP options set** is `ec2.internal` in region `us-east-1`, and `region.compute.internal` in all other regions. If you enter a domain name value in the **DHCP options set** (typically something like `.mycompany.com`), the value you enter will be used, instead.
 - If the `enableDnsHostnames` and `enableDnsSupport` settings in the VPC both are true, then the domain name will be the default value, either `ec2.internal` in region `us-east-1` or `region.compute.internal` in all other regions.

Using the hostname example above, a typical DNS name would be:

```
ip-10-1-2-3.mycompany.com
```

SSH Host Key Retrieval and Verification

When Altus Director logs into an instance through SSH, by default it does not perform host key verification against the remote host. For added security and to prevent man-in-the-middle attacks, host key verification can be enabled by setting the host key fingerprint retrieval type.

The `sshHostKeyRetrievalType` can be set in the conf file to specify the desired host key fingerprint retrieval type. The allowable values are: NONE, PROVIDER, INSTANCE, and FALLBACK.

NONE

Altus Director will not attempt to retrieve the host key fingerprints for the instances, and host key verification is not performed. This is the default behaviour.

PROVIDER

Altus Director will attempt to retrieve the host key fingerprints for each instance from the cloud provider. This is currently only supported for AWS. This is done in AWS by using the API to read the console output for the instance, which usually displays the host key fingerprints associated with the instance. See [Getting Console Output and Rebooting Instances](#) in the AWS documentation for more information on reading the console output. Since Altus Director has to wait for the console output to appear, this will increase bootstrap time by around three to five minutes.

Note that not all AMIs display the host key fingerprints for the instance in the console output. Before enabling this, ensure that the AMI displays a section that looks like the following in the console output:

```
ec2: -----BEGIN SSH HOST KEY FINGERPRINTS-----
ec2: 1024 6d:99:6d:f1:d5:42:42:68:f1:5b:40:e9:ff:30:82:38 /etc/ssh/ssh_host_dsa_key.pub
   (DSA)
ec2: 2048 2d:e1:d3:48:06:0d:32:32:1b:14:3a:87:49:18:ca:2a /etc/ssh/ssh_host_key.pub
   (RSA1)
ec2: 2048 7f:1b:3b:51:42:2e:4e:be:9f:f1:77:15:a6:33:62:c7 /etc/ssh/ssh_host_rsa_key.pub
   (RSA)
ec2: -----END SSH HOST KEY FINGERPRINTS-----
```

INSTANCE

On the first SSH connection into the instance, Altus Director will retrieve and store the host key fingerprints by reading the host key files in `/etc/ssh`. Future SSH connections will be verified using the stored fingerprints. This method is less secure than the `PROVIDER` method, since the very first SSH connection isn't verified.

FALLBACK

This approach is a combination of `PROVIDER` and `INSTANCE`. Altus Director will first attempt to retrieve the host key fingerprints from the cloud provider (using `PROVIDER` method). If that fails or is not supported, it will attempt to retrieve the fingerprints through the instance on the first SSH connection (`INSTANCE` method).

Creating Kerberized Clusters With Altus Director

Using Altus 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 Altus Director client, using the `bootstrap-remote` command to send the configuration file to a running Altus Director server.

You must have an existing Kerberos Key Distribution Center (KDC) set up, and it must be reachable by the instance where Altus 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 Altus Director. Kerberos must be enabled through Altus Director using the configuration file, at the time of cluster setup.

Creating a Kerberized Cluster with the Altus Director Configuration File

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

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

Configuration setting	Description
krbAdminUsername	An administrative Kerberos account with permissions that allow the creation of principals on the KDC that Cloudera Manager will be using. This is typically in the format <i>principal@your.KDC.realm</i>
krbAdminPassword	The password for the administrative Kerberos account.
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 <code>true</code> . This allows Cloudera Manager to deploy Kerberos configurations to cluster instances. The value <code>false</code> is not supported for this configuration setting.
KRB_ENC_TYPES	The encryption types your KDC supports. Some of encryption types listed in the sample configuration file require the unlimited strength JCE policy files.

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

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

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

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

unlimitedJce: true
# Kerberos principal and password for use by Altus 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"
  }
}
}
```

Enabling Sentry Service Authorization

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

Prerequisites

- Altus 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 Altus Director CLI

For this method, you use the Altus Director client and the `bootstrap-remote` command to send a configuration file to the Altus 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 Altus 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 Altus Director API

You can use the Altus 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](#).

Creating Highly Available Clusters With Altus Director

Using Altus 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 Altus Director configuration file as described in this topic and launch the cluster with the Altus Director client and the `bootstrap-remote` command, which sends the configuration file to a running Altus Director server.



Note: With Altus 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 [Altus Director GitHub site](#).

Limitations and Restrictions

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

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

Editing the Configuration File to Launch a Highly Available Cluster

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

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



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

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

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

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

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

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

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

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

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

- YARN

- Two RESOURCEMANAGER roles.
- One JOBHISTORY role.

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

- ZooKeeper

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

- HBase

- Two MASTER roles.

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. Altus 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 Altus Director. If no databases are specified in the configuration file, an embedded PostgreSQL database is used.

- External databases can be set up by Altus Director, or you can configure preexisting external databases to be used. Databases set up by Altus 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 Altus 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 Altus Director, satisfies this requirement.

Migrating HDFS Master Roles

Situations can arise in a cluster where you need to migrate HDFS master roles, perhaps as a result of hardware failure or because of a need to rebalance cluster resources. This section describes two methods of migrating HDFS master roles (NameNode, Failover Controller, and JournalNode) in a highly available cluster:

- **Migrating roles to a different instance:** Cloudera recommends using this method because it gives you the most options:
 - It allows you to move the master roles from one instance to another without necessarily removing the instance that you are migrating from.
 - It allows you to make changes to the instance configuration, such as the AMI, security group, or instance type, rather than requiring you to move the master roles to an instance with a configuration identical to the one being replaced.
- **Replacing a HDFS master instance with an identical copy:** With this method, you replace the original instance with the new instance, creating an identical copy without changes to the instance configuration. At the end of the procedure, the original instance is deleted.

With either method, you will perform a process that is partly manual, and requires migration of the roles in Cloudera Manager. If a host running HDFS master roles fails in a highly available cluster, you can use Altus Director and the Cloudera Manager Role Migration wizard to move the roles to another host without losing the role states. What was previously the 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.

The procedure for each method is described in the following sections.

Migrating Roles to a Different Instance

To migrate master roles from one instance to another, follow the steps below:

Step 1: Preparation

1. Back up the Altus Director database.
2. Check the logs to ensure that the cluster refresh is working properly, since cluster refresh is required for this procedure. The cluster refresh process needs access to both Cloudera Manager and the cloud provider. Misconfiguration that prevents this access might appear in the log file. Look for log messages that contain `ClusterRefresher` and `RefreshClusters` to ensure that no warnings or errors appear. Altus Director server logs are located at `/var/log/cloudera-director-server/application.log` on the Altus Director instance.

Step 2: Add instances (without roles) to the cluster in Altus Director

1. Create a new instance group for each set of roles you will place on new instances in [Step 3: Migrate roles to the new instances in Cloudera Manager](#) on page 162. Typically, for HDFS master roles you will create three instances for the HDFS master roles, in two instance groups:
 - Two of the instances will include all three HDFS master roles (NameNode role, Failover Controller role, and JournalNode role). Create one instance group for these three roles.
 - The third instance will include the required third copy of the JournalNode role. Create a second instance group for this JournalNode role.

For information about configuring instance groups in Altus Director, see [Configuring Instance Groups During Cluster Creation](#) on page 210.

2. In Altus Director, add the required number of new instances (the number of instances you are replacing), but do not assign roles to them. Altus Director installs the Cloudera Manager agent on these instances, but no roles, so that they are available as hosts for Cloudera Manager to use in the next step, [Step 3: Migrate roles to the new instances in Cloudera Manager](#) on page 162. For information about how to add instances in Altus Director, see [Modifying the Number of Instances in an Existing Cluster](#) on page 211.
3. If you want to change the configuration of the new master instances, create new instance templates in the environment that includes your cluster. Configure these instance templates with the desired configurations for the instances you are replacing. If you want the new instances to be identical to the ones you are replacing, you can use the instance templates that were used to create the original instances.

Step 3: Migrate roles to the new instances in Cloudera Manager

In Cloudera Manager, migrate roles to the new instances. Refer to the [Cloudera Manager and CDH documentation](#) for instructions on migrating high availability master role instances.

Step 4: Wait for the Altus Director refresh to pick up the new role assignments

The Altus Director refresh runs approximately every five minutes. You can monitor the log file for the log messages for `ClusterRefresher` and `RefreshClusters` to see when this occurs.

Verify that the high availability master roles are as expected in the Altus Director UI. The old instances should contain no roles.

Step 5: Terminate old instances in Altus Director

In Altus Director, terminate the old instances. For information on terminating instances in Altus Director, see [Modifying the Number of Instances in an Existing Cluster](#) on page 211.

Replacing a HDFS Master Instance with an Identical Copy

Use this method to replace instances with an exact copy on the new instance. If you want to make changes in the instance or its configuration, for example, to use a different instance type or an instance with more memory, use the workflow described above, [Migrating Roles to a Different Instance](#) on page 161.

The following limitations apply when performing HDFS role migration with this method.

- You cannot modify any instance groups on the cluster during the repair and role migration process.
- You cannot clone the cluster during the repair and role migration process.
- During role migration with this method, you cannot perform Altus Director operations on the cluster, such as growing or shrinking it.
- [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.



Important: 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 Altus Director. Once the old instance is terminated, any information or state it contained is lost.

If you have completed Step 1 (on Altus 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 Altus Director and clicking either the link for migration in the upper left, or the **Modify Cluster** button on the right. A pop-up displays the hosts to migrate from and to:

Manual Role Migration

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

Migrate the roles by using Cloudera Manager commands 

I have manually migrated these roles

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

Ignore for now, I will complete manual role migration later

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: Create a new instance in Altus Director

1. In Altus 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 collocated 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.

Altus 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: Migrate roles and data in Cloudera Manager

In Cloudera Manager, open the cluster. 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: Delete the old instance in Altus Director

1. In Altus Director, return to the cluster.
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.

Encrypted Configuration Properties

To protect sensitive data such as password, you can provide configuration properties for the Altus Director server and client in an encrypted form. This topic describes how to use encrypted configuration properties.

Creating an Encrypted Configuration Property

Use the `encryptPropertyValues` script included with the Altus Director server to generate encrypted property values. As arguments to the script, pass an encryption password and one or more values to encrypt. Encryption parameters are taken from the server's current configuration.

```
$ ./encryptPropertyValues master-password value1 value2 value3
Encrypting 'value1' with algorithm PBEWITHSHA1ANDDESEDE ...
-----
1LC5sbvMlv0I0e88HmCmvA==
Encrypting 'value2' with algorithm PBEWITHSHA1ANDDESEDE ...
zWgV8NvRITwJVZvqsuvYSw==
-----
Encrypting 'value3' with algorithm PBEWITHSHA1ANDDESEDE ...
xjUEn2cHVzozR+0JhO+/bg==
-----
```

To use an encrypted value, provide it as the value for a configuration property through any supported mechanism, such as in the `application.properties` file. The value must be wrapped in parentheses and preceded by the string `ENC` to indicate that it is an encrypted value.

```
property1: ENC(1LC5sbvMlv0I0e88HmCmvA==)
property2: ENC(zWgV8NvRITwJVZvqsuvYSw==)
property3: ENC(xjUEn2cHVzozR+0JhO+/bg==)
```

Using Encrypted Configuration Properties

When encrypted configuration properties are in place, Altus Director must be started with the password originally used to encrypt them. This lets Altus Director decrypt the values.

The password is provided to Altus Director through the `jasypt.encryptor.password` configuration property. While this property could itself be provided in `application.properties`, that would defeat the protection provided by encryption. Better alternatives:

- Pass the value on the command line when starting Director. This option, however, reveals the password in lists of active commands from OS utilities like `ps`.

```
$ ./bin/start --jasypt.encryptor.password=master-password
```

- Set the `JASYPT_ENCRYPTOR_PASSWORD` environment variable.

```
$ export JASYPT_ENCRYPTOR_PASSWORD=master-password
$ ./bin/start
```

- Set the `jasypt.encryptor.password` Java system property. Doing this requires editing of the `cloudera-director-server` and `cloudera-director` scripts.

For maximum protection, store the password in a key management service, such as those supplied by cloud providers, and retrieve the password whenever you start Altus Director.

Configuring Encryption of Configuration Properties

Several Altus Director configuration properties control the type of encryption used, such as the algorithm and number of rounds. Set these properties in the Altus Director `application.properties` file as desired, and then use the `encryptPropertyValues` script to encrypt new property values using them. Caveats:

- When you change the encryption configuration, regenerate all encrypted configuration values.
- Some encryption algorithms and parameters require use of unlimited strength Java security policy files to function. The default choices used by Altus Director do not.
- Some encryption algorithms require installation of a new security provider, such as [Bouncy Castle](#).
- Altus Director might not be able to decrypt values encrypted with an algorithm that uses an HMAC because the ciphertext currently does not include the necessary initialization vector (IV).

Important configuration properties for configuration property encryption are in the following table.

Configuration Property	Default	Meaning
<code>jasypt.encryptor.algorithm</code>	PBEWITHSHA1ANDDESEDE	encryption algorithm
<code>jasypt.encryptor.keyObtentionIterations</code>	10000	number of rounds
<code>jasypt.encryptor.providerName</code>	SunJCE	name of security provider supporting encryption algorithm
<code>jasypt.encryptor.saltGeneratorClassname</code>	<code>org.jasypt.salt.RandomSaltGenerator</code>	fully-qualified name of Java class that generates salt for encryption
<code>jasypt.encryptor.stringOutputType</code>	base64	encoding for ciphertext

Configuring and Running Altus Director

The topics in this section explain how to configure and run Altus Director.

Auto-Repair for Failed or Terminated Instances

Altus 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 133.



Note: Do not use auto-repair on Microsoft Azure with Altus Director 2.6 and higher. Healthy Azure instances can sometimes be removed when there is a connection failure between Altus Director and Azure.

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:

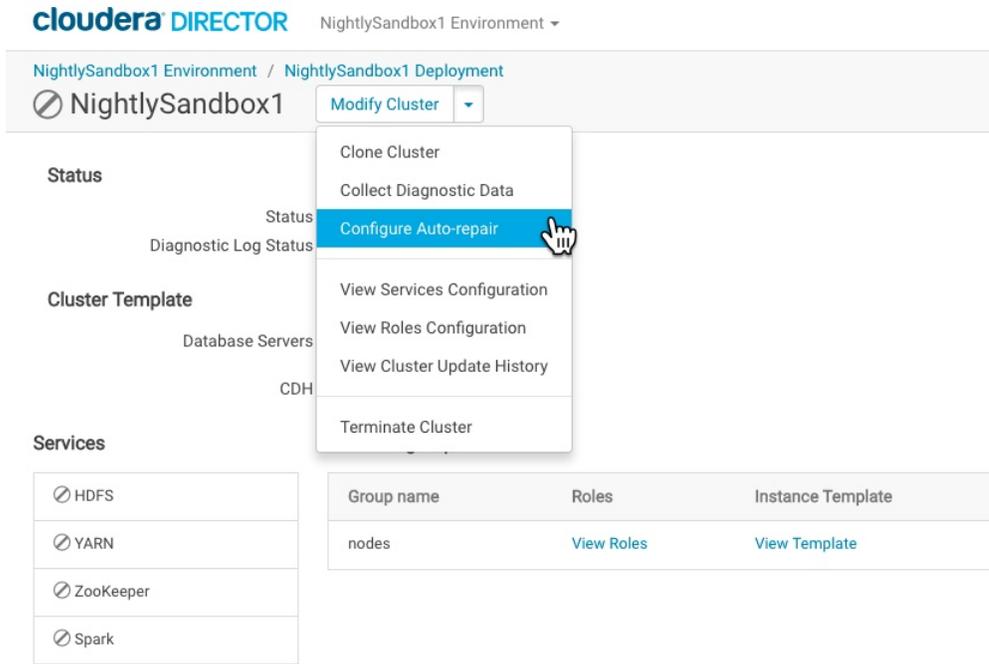
- Services**
- Core Hadoop**
HDFS, Hive, Hue, Oozie, YARN, ZooKeeper
 - Core Hadoop with HBase**
HBase, HDFS, Hive, Hue, Oozie, YARN, ZooKeeper
 - Core Hadoop with Impala**
HDFS, Hive, Hue, Impala, Oozie, YARN, ZooKeeper
 - Core Hadoop with Search**
HDFS, Hive, Hue, Oozie, Solr, YARN, ZooKeeper
 - Core Hadoop with Spark on YARN**
HDFS, Hive, Hue, Oozie, Spark on YARN, YARN, ZooKeeper
 - Real Time Ingest**
Flume, Kafka, ZooKeeper
 - All Services**
Flume, HBase, HDFS, Hive, Hue, Impala, Kafka, Key-Value Store Indexer, Oozie, Solr, Spark on YARN, YARN, ZooKeeper

Enable Auto-repair ?

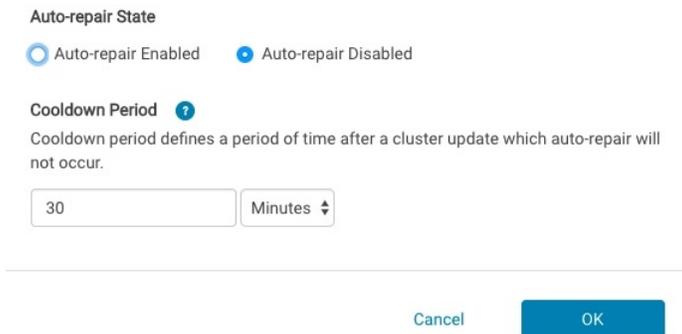
Instance groups

Group name ?	Roles	Instance Template	Instance C
masters	Edit Roles	Select a Template ⌵	
workers	Edit Roles	Select a Template ⌵	
gateway	Edit Roles	Select a Template ⌵	

- 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 Altus 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 Altus Director 2.5 and higher running with Cludera 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 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, Altus 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.

Deploying Java on Cluster Instances

When you set up Cloudera Manager and CDH clusters in the cloud, a version of the Java JDK must be installed on each instance. Choose one of the three JDK installation strategies described below for your Cloudera Manager instance and the CDH clusters it manages. The JDK installation strategy can be set using a configuration file or using the Altus Director API, but it is not currently configurable in the Altus Director web UI.

Once a particular JDK installation strategy has been used to bootstrap a new Cloudera Manager deployment and JDK cluster, Altus Director continues to follow that JDK installation strategy for all additional clusters added to that Cloudera Manager deployment.



Note: Cloudera Manager 6 and CDH 6 require JDK version 8.

AUTO JDK Installation Strategy

With the AUTO setting, Altus Director installs the JDK on the Cloudera Manager node, and then Cloudera Manager handles JDK installation for all instances in the cluster. If needed, Altus Director also installs unlimited strength JCE policy files on the Cloudera Manager instance, and directs Cloudera Manager to install them on all cluster instances.

The JDK version installed depends on the `cmJavaPackages` and `defaultCmJavaPackage` properties in the `application.properties` file, as described below:

- By default, Altus Director 6 installs Java 8 for Cloudera Manager 6 on all nodes. By default, Altus Director 6 installs Java 7 for Cloudera Manager 5 on all nodes.
- When Altus Director can determine the Cloudera Manager version, the configuration property `lp.bootstrap.packages.cmJavaPackages` determines the Java package used. The property is a list of key-value pairs, where each key is a regular expression for a Cloudera Manager version, and each value is the corresponding Java package to use for it. In this example, Java 8 is used for Cloudera Manager 5.13.x, and Java 7 for any other version:

```
5\.13\.\.*=oracle-j2sdk1.8,5\.\.*=oracle-j2sdk1.7
```

- When Altus Director cannot determine the Cloudera Manager version, the configuration property `lp.bootstrap.packages.defaultCmJavaPackage` determines the Java package used.

To guarantee that JDK 8 is used, change `cmJavaPackages` to `.*=oracle-j2sdk1.8` and change `defaultCmJavaPackage` to `oracle-j2sdk1.8`.

You must ensure that yum is configured with a repository that contains the Java package you have chosen, as described below:

- If you select `oracle-j2sdk1.8` for Cloudera Manager 6, which is the default behavior for Altus Director, then you don't have to do anything further, because Altus Director always configures the Cloudera Manager repository, and the Cloudera Manager repository has `oracle-j2sdk1.8`.
- If you select `oracle-j2sdk1.8` for Cloudera Manager 5, then you need to use a bootstrap script or AMI that ensures that the right yum repository is configured. The repository for Altus Director 2.x has `oracle-j2sdk1.7` and `oracle-j2sdk1.8`.

Configuring and Running Altus Director

AUTO is the default JDK installation strategy, so it is not necessary to specify it in a configuration file or using the API. All installations done with the Altus Director web UI use the default AUTO setting.

DIRECTOR_MANAGED JDK Installation Strategy

With the `DIRECTOR_MANAGED` setting, Altus Director installs the JDK for all Cloudera Manager and cluster instances from a yum repository. The JDK installation includes unlimited strength JCE policy files, should you require them.

With this installation strategy, you can choose the Java version to be installed for Cloudera Manager and for CDH cluster instances.

- By default, Altus Director 6 installs Java 8 for Cloudera Manager 6 on all nodes. By default, Altus Director 6 installs Java 7 for Cloudera Manager 5 on all nodes.
- When Altus Director can determine the Cloudera Manager version, the configuration property `lp.bootstrap.packages.cmJavaPackages` determines the Java package used. The property is a list of key-value pairs, where each key is a regular expression for a Cloudera Manager version, and each value is the corresponding Java package to use for it. In this example, Java 8 is used for Cloudera Manager 5.13.x, and Java 7 for any other version:

```
5\.13\.\.*=oracle-j2sdk1.8,5\.\.*=oracle-j2sdk1.7
```

- When Altus Director cannot determine the Cloudera Manager version, the configuration property `lp.bootstrap.packages.defaultCmJavaPackage` determines the Java package used.

To guarantee that JDK 8 is used, change `cmJavaPackages` to `.*=oracle-j2sdk1.8` and change `defaultCmJavaPackage` to `oracle-j2sdk1.8`.

You must ensure that yum is configured with a repository that contains the Java package you have chosen, as described below:

- If you select `oracle-j2sdk1.8` for Cloudera Manager 6, then you don't have to do anything further, because Altus Director always configures the Cloudera Manager repository, and the Cloudera Manager 6 repository has `oracle-j2sdk1.8`.
- If you select `oracle-j2sdk1.8` for Cloudera Manager 5, then you need to use a bootstrap script or AMI that ensures that the right yum repository is configured. The repository for Altus Director 2.x has `oracle-j2sdk1.7` and `oracle-j2sdk1.8`.

Here is how the `DIRECTOR_MANAGED` setting looks in a configuration file:

```
...
cloudera-manager {
    instance: ${instances.m3x} {
        tags {
            application: "Cloudera Manager 5"
        }
    }
    javaInstallationStrategy: DIRECTOR_MANAGED
    ...
}
```

NONE JDK Installation Strategy

With the `NONE` installation strategy, neither Altus Director nor Cloudera Manager install a JDK; you must manage all JDK installation yourself, for example, by using bootstrap scripts or pre-baked images. When running a bootstrap script, Altus Director can install any Java package that instances can find with yum.

Here is how this setting looks in a configuration file:

```
...
cloudera-manager {
    instance: ${instances.m3x} {
```

```

    tags {
      application: "Cloudera Manager 5"
    }
  }
  javaInstallationStrategy: NONE
  ...
}

```

Setting Altus Director Properties

This topic lists the configuration properties recognized by Altus 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 Altus Director features you want to use.

Setting Configuration Properties

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

```
./bin/cloudera-director bootstrap-remote 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 client) or the `/etc/cloudera-director-server/application.properties` (for the server) in the Altus Director installation. Any changed properties in these files take effect automatically when Altus Director is restarted. To override existing properties, set new values in the command line.

While the `application.properties` files are located on the Altus 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 Altus Director

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

- **RHEL and CentOS** - When new properties are introduced in Altus Director, they are added to `application.properties.rpmnew`. The original `application.properties` file functions as before and is not overwritten with the new Altus 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 Altus 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 Altus 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

Type	Description
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 Altus Director Instances

The Altus 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 Altus Director Server

Although you can stop and start the Altus 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.*

Pausing a cluster also requires that the private IP address for all Azure Virtual Machines be set to static.

Notes on Pausing a Cluster



Important: 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 Altus Director, you should confirm that the IP configuration of your hosts is set to **static** before shutting down your cluster. The default for the Altus Director Azure plugin is **dynamic**. If you need to change the IP configuration from **dynamic** to **static** it can be done via the [Azure Portal](#) or [PowerShell](#).



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. Change all virtual machine's private IP address from dynamic to static by following the instructions in the **Important** note paragraph above in [Notes on Pausing a Cluster](#).
2. Stop the cluster
3. Stop Cloudera Management Service
4. 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.

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.

Configuring Altus Director Server for LDAP and Active Directory

By default, the Altus Director server uses its own internal database to store user accounts and authorizations. You can instead configure the server to perform authentication and authorization using an external LDAP server, including Active Directory.



Note: This section describes configuring Altus Director itself for use of LDAP, and not configuring any installations of Cloudera Manager or CDH clusters that Altus Director bootstraps.

User and Group Model

When configured for LDAP, Altus Director expects that each user has an entry in an LDAP server under some base DN. When a user attempts to log in to the Altus Director server, the server will locate the user in LDAP and try to authenticate the user against the LDAP server using the provided password.

Altus Director expects that LDAP groups, collected under some base DN, are used to determine the roles that a user is authorized for. If a user is a member of a group in LDAP, then the user is granted the role that maps to that group. Multiple groups can map to the same role in Altus Director, and users can have multiple roles. Altus Director does not support the use of nested groups in LDAP to determine roles.

Basic LDAP Configuration

Determine the following pieces of information in order to configure Altus Director for LDAP. The same information is also needed for Active Directory.

- The LDAP server host and port.
- The bind DN and password that Altus Director should use when searching for users and groups. This account does not need administrative access to the LDAP server, but only read access for the necessary searches.
- The base DN for user searches.

Configuring and Running Altus Director

- The filter to use for user searches or the DN pattern that all user DNs adhere to. More information on these is provided below.
- The base DN for group searches.
- The filter to use for group searches. More information on this is provided below.
- The attribute of each group that forms the basis for an Altus Director role. More information on this is provided below.

The user search filter is employed to locate a user attempting to log in to Altus Director. The user's login username is substituted into the filter before it is submitted to the LDAP server. The string `{0}` indicates where the username is substituted.

- To look for a username in the **uid** attribute of a user entry, use the filter `(uid={0})`.
- To look for a username in the **cn** attribute of a user entry, use the filter `(cn={0})`.
- For Active Directory, use the filter `(sAMAccountName={0})`.

Instead of specifying a user search filter, you can instead supply a user DN pattern. This avoids a search, because Altus Director can determine a user DN by substituting the user's login username into the pattern. The string `{0}` indicates where the username is substituted.

- The user DN pattern cannot be used for Active Directory. Instead, a user search filter must be supplied.

The group search is employed to locate groups that an authenticated user belongs to. The user's DN is substituted into the filter before it is submitted to the LDAP server. The string `{0}` indicates where the username is substituted.

- When using the objectclass **groupOfUniqueNames** for user groups, use the filter `(uniqueMember={0})`.
- For Active Directory, use the filter `(member={0})`.

The group role attribute selects which attribute of a group entry is used as the basis for determining the corresponding Altus Director role. Usually this is the CN of the group, but any attribute can be selected instead.

- The group role attribute is not used for Active Directory. The group name is always used as the basis for determining a role.

The following configuration properties control basic LDAP connectivity. Additional configuration properties described in a later section are needed for Active Directory.

Configuration Property	Default	Meaning
<code>lp.security.userSource</code>	internal	Where to look for user data; for LDAP or Active Directory connectivity, set this to LDAP .
<code>lp.security.ldapConfig.url</code>	N/A	The LDAP URL, with host and optional port, e.g.: <code>ldap://ldaphost:389/</code> .
<code>lp.security.ldapConfig.bindDn</code>	N/A	The bind DN for Altus Director to use for searches.
<code>lp.security.ldapConfig.bindPw</code>	N/A	The bind password for Altus Director to use for searches.
<code>lp.security.ldapConfig.userSearchBase</code>	N/A	The base DN for user searches.
<code>lp.security.ldapConfig.userSearchFilter</code>	N/A	The user search filter, e.g., <code>(uid={0})</code> .
<code>lp.security.ldapConfig.ldapDnPattern</code>	N/A	The DN pattern for users.
<code>lp.security.ldapConfig.groupSearchBase</code>	N/A	The base DN for group searches.
<code>lp.security.ldapConfig.groupSearchFilter</code>	N/A	The group search filter, e.g., <code>(uniqueMember={0})</code> .
<code>lp.security.ldapConfig.groupRoleAttribute</code>	N/A	The group attribute to use as a basis for selecting a role, e.g., cn .

After setting or changing these configuration properties, restart the Altus Director server for them to take effect.

The following is an example of an LDAP configuration that points to an OpenLDAP server. Each user's **uid** attribute is used as their login username, and group membership is determined by the **uniqueMember** attribute.

```
lp.security.userSource: LDAP
lp.security.ldapConfig.url: ldap://openldaphost/
lp.security.ldapConfig.bindDn: cn=ldapadm,dc=domain,dc=example
lp.security.ldapConfig.bindPw: password
lp.security.ldapConfig.userSearchBase: ou=People,dc=domain,dc=example
lp.security.ldapConfig.userSearchFilter: (uid={0})
lp.security.ldapConfig.ldapDnPattern: uid={0},ou=People,dc=domain,dc=example
lp.security.ldapConfig.groupSearchBase: ou=Groups,dc=domain,dc=example
lp.security.ldapConfig.groupSearchFilter: (uniqueMember={0})
lp.security.ldapConfig.groupRoleAttribute: cn
```

Local User Management under LDAP

Altus Director does not provide user management services when configured for LDAP. To make changes to user accounts and roles, make the necessary changes in the source LDAP server. Users might need to log out of Altus Director and log back in for changes to take effect.

Most user API endpoints for the Altus Director server are disabled when the server is configured for LDAP, and will return the HTTP response code 400 (Bad Request) for any request. Also, the Altus Director server UI does not present user management capabilities, such as password changes, when the server is configured for LDAP.

Role Mapping

Altus Director maps LDAP groups to its own roles in a simple fashion.

1. The value of the group role attribute, such as **cn**, is found.
2. The value is converted to uppercase.
3. The prefix **ROLE_** is prepended.

Therefore, a group with a role attribute of **admin** is converted to the role **ROLE_ADMIN**. When using a role attribute of **cn**, then the DN for the group is similar to **cn=admin,ou=Groups,dc=domain,dc=example**.

Currently, the only roles supported by Altus Director are:

- **ROLE_READONLY** - read-only / guest access
- **ROLE_ADMIN** - full access

An administrative account should have all roles, and therefore be a member of all mapped groups.

Role mapping is more complex for Active Directory and is described below.

Active Directory Configuration

Determine the following additional pieces of information in order to configure Altus Director for Active Directory.

- The domain for users and groups.
- The desired role mapping from group names to Altus Director roles.

The following configuration properties control Active Directory connectivity, in addition to those for basic LDAP configuration.

Configuration Property	Default	Meaning
lp.security.ldapConfig.activeDirectory.domain	N/A	The domain.
lp.security.ldapConfig.activeDirectory.roleMapping.*	N/A	The role mapping (multiple properties).

After setting or changing these configuration properties, restart the Altus Director server for them to take effect.

Configuring and Running Altus Director

The following is an example of an LDAP configuration that points to an Active Directory server. Each user's **sAMAccountName** attribute is used as their login username, and group membership is determined by the **member** attribute.

```
lp.security.userSource: LDAP
lp.security.ldapConfig.url: ldap://adhost/
lp.security.ldapConfig.bindDn: cn=Administrator,ou=Users,dc=domain,dc=example
lp.security.ldapConfig.bindPw: password
lp.security.ldapConfig.userSearchBase: ou=Users,dc=domain,dc=example
lp.security.ldapConfig.userSearchFilter: (sAMAccountName={0})
lp.security.ldapConfig.groupSearchBase: ou=Groups,dc=domain,dc=example
lp.security.ldapConfig.groupSearchFilter: (member={0})
lp.security.ldapConfig.activeDirectory.domain: DOMAIN.EXAMPLE
lp.security.ldapConfig.activeDirectory.roleMapping.adminGroup: admin
lp.security.ldapConfig.activeDirectory.roleMapping.readonlyGroup: readonly
```

Role Mapping

Altus Director provides flexibility in mapping the names of Active Directory groups to Altus Director roles. The mapping process is:

1. The group name is searched for among the roleMapping configuration properties.
 - a. If one is found, the property value is used as the base name for the role.
 - b. If one is not found, the group is ignored.
2. The base name is converted to uppercase.
3. The prefix **ROLE_** is prepended.

In the example below, two groups named **group1Name** and **group2Name** are mapped to the **ROLE_READONLY** Altus Director role, while another **group3Name** is mapped to the **ROLE_ADMIN** role.

```
lp.security.ldapConfig.activeDirectory.roleMapping.group1Name: readonly
lp.security.ldapConfig.activeDirectory.roleMapping.group2Name: readonly
lp.security.ldapConfig.activeDirectory.roleMapping.group3Name: admin
```



Note: If no role mapping is provided in the configuration, then no roles are granted to authenticated users by the Altus Director server, and they will be unable to use it.

Configuring Altus Director for a New AWS Instance Type

Amazon Web Services occasionally introduces new instance types with improved specifications. Altus 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. Altus 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 Altus 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 Altus Director 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 virtualization type.

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

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

Updated Ephemeral Device Mappings

Each AWS instance type provides zero or more instance store volumes, also known as ephemeral storage. These volumes are distinct from EBS-backed storage volumes; some instance types include no ephemeral storage. Altus 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 Altus 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 Altus Director 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 Altus Director server so that they are detected and overlaid on the built-in mappings.

Configuring and Running Altus Director

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

Configuring Altus Director to Use Custom Tag Names on AWS

In addition to user-specified tags, Altus 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 Altus 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 Altus Director uses can 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 Altus Director server so that they are detected and used.



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

Using Altus Director Server to Manage Cloudera Manager Instances

The Altus 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. Altus Director server configuration and use are described in the following topics.

Altus Director and Cloudera Manager Usage

Altus 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 Altus Director, you can monitor and manage multiple Cloudera Manager and CDH deployments, across different cloud environments.

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

- With Altus Director 2.4 and higher and Cloudera Manager and CDH 5.11 and higher, many changes made directly in Cloudera Manager are detected by Altus Director, which periodically refreshes its state to reflect the state of the cluster in Cloudera Manager. Altus 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 Altus Director to refresh before making any cluster modifications in Altus Director-side cluster modifications, such as grow or shrink.

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

When to Use Altus Director

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

CDH Cluster Management Tasks

When you deploy CDH and Cloudera Manager through Altus Director, you use Altus 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 Altus Director and Cloudera Manager:

Task	Application	Notes
Cluster setup	Altus Director	Altus 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	Altus Director	
Host decommission	Altus Director	This is done by deleting the instance from the virtual instance group using Altus Director.
Adding a service	Cloudera Manager	If you add a service to a cluster in Cloudera Manager, Altus 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, Altus 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 Altus Director to create hosts that do not have that service's roles.
Initial role assignment	Altus Director	
Add new virtual instance groups to a cluster	Altus Director	
Change role assignments for an existing virtual instance group, or migrate roles from one instance to another	Cloudera Manager	Altus Director periodically refreshes its data on the state of cluster roles in existing virtual instance groups to include changes made with Cloudera Manager. 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 182 below for more information.
Changes to the configuration of a service or role	Cloudera Manager	Altus 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 182 below for more information.
Cloud provider settings for instances, such as the machine image or instance type	Cloud provider management tool	Altus 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

Task	Application	Notes
		instance group. In AWS, for example, you can change the AMI or instance type directly in the AWS management console, but you must change all instances in the instance group to be identical. See Ensuring Consistency of Virtual Instance Groups on page 182 below for more information.
Adding, removing, and modifying parcels	Cloudera Manager	If you activate or deactivate parcels in Cloudera Manager, Altus 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 Altus 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 Altus Director. If you do not update the information in Altus Director, Altus 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. Altus 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. Altus Director will detect the version changes, and new clusters will use the upgraded versions.
Enabling high availability during cluster setup	Altus Director	High availability is supported in Altus Director version 2.0 or higher. Use the configuration file to enable high availability. Do not use the Altus 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. <div style="border: 1px solid black; padding: 5px;"> <p> Note: When Altus 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 Altus Director in sync. This is not an issue if you are using Cloudera Manager 5.12 and higher.</p> </div>
Modifying a cluster in a highly available deployment	Altus 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	Altus Director	Kerberos setup is supported in Altus Director version 2.0 or higher. Use the configuration file to enable Kerberos during cluster setup. After the cluster is created, the configuration file can no longer be used to enable Kerberos. Do not use the Altus Director web UI. If you use Altus Director to deploy a cluster but use Cloudera Manager to enable Kerberos authentication, Altus 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 Altus Director. When using Altus Director 2.4 and above with Cloudera Manager and CDH 5.11 and above, you can change role assignments, role configurations, and cloud provider settings in Cloudera Manager, but you must ensure that all instances in a given virtual instance group are configured identically. Altus 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, Altus Director will detect the inconsistency and will flag the virtual instance group in the Altus 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 Altus Director, shrink the virtual instance group to remove the instances that are flagged as inconsistent

CDH Cluster Management Guidelines for Altus Director

When you use Altus Director to deploy Cloudera Manager and CDH, the cluster information is saved in the Altus Director database. If you make changes to the cluster using the cloud provider management console, the changes are detected by Altus Director. But terminating an instance using the cloud provider management console results in poor health of the hosts and services in Altus Director. If the health of an instance turns bad or the instance fails, you can migrate to a new instance. Use the Altus Director web UI to shrink and grow the worker nodes and migrate the master node to a new instance.

For information about growing or shrinking a cluster, see [Modifying the Number of Instances in an Existing Cluster](#) on page 211.

For information about migrating HDFS master roles to a new instance, see [Migrating HDFS Master Roles](#) on page 161.

Setting Cloudera Manager Configurations

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

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

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

- **Client** - Using the configuration file.
- **Server** - Using the Altus 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 Altus 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 Altus Director API calls and in the configuration name fields of the Altus 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 5.14.0 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**.

Altus 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 Altus Director-managed clusters using Cloudera Manager directly. For these use cases, Altus Director will sync up automatically with changes made in Cloudera Manager. Other

Using Altus Director Server to Manage Cloudera Manager Instances

configuration changes cannot be safely made using Cloudera Manager directly because Altus 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 [Altus Director and Cloudera Manager Usage](#) on page 179.

Setting up a Cloudera Manager License

There are three ways to set up a Cloudera Manager license using Altus 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 `Licensing` 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"  
        }  
      }  
    }  
  }  
}
```

Ports Used by Altus Director

Altus 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 Altus 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 Altus 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	<i>security_group_id</i>
SSH (22)	TCP (6)	22	0.0.0.0/0

In a restricted network environment, you might 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 Altus Director server. If using SSH tunneling, the other Altus Director ports below are not required.
- Minimally, the Altus 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 Altus Director server to enable access to the Altus Director web UI. Optionally, you can configure Altus Director to use HTTPS. You can configure a non-default port for the Altus 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 Altus Director server can use port 7180 to interact with the Cloudera Manager API. (Otherwise, Altus Director will use SSH tunnels on port 22 to communicate with Cloudera Manager.)
- The Altus 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 Altus Director port requirements described above:

Service	Role	Purpose	Default Port	Protocol	Required?
Altus Director	Altus Director server	Altus 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)
Clusters managed by Altus 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

Service	Role	Purpose	Default Port	Protocol	Required?
	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.	Altus 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 Altus 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 can 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
```

Altus Director's Use of SSH Keys

Altus Director uses SSH to issue commands to instances that it launches and configures as part of bootstrapping deployments and clusters. Therefore, Altus 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 Altus 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 Altus 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, can override the username.

The private key must be provided to Altus Director in unencrypted form. However, Altus Director encrypts its own database where it stores private keys.

Good Practices for SSH Key Management with Altus 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 Altus Director database encryption password.** Altus 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 Altus Director. This allows the lifecycle of personal and service key pairs to be managed separately. Also, service key pairs can 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.

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, Altus Director requires permissions for the following methods:
 - CreateTags
 - DescribeAvailabilityZones
 - DescribeImages
 - DescribeInstanceStatus
 - DescribeInstances
 - DescribeKeyPairs
 - DescribePlacementGroups

Using Altus Director Server to Manage Cloudera Manager Instances

- DescribeRegions
 - DescribeSecurityGroups
 - DescribeNetworkAcls
 - DescribeSubnets
 - DescribeInstanceAttribute
 - RunInstances
 - TerminateInstances
- To use SSH host key retrieval type with the PROVIDER option, the following additional EC2 permission is required:
 - GetConsoleOutput
- For more information, see [SSH Host Key Retrieval and Verification](#).
- To use EBS volumes, the following additional EC2 permissions are required:
 - CreateVolume
 - DescribeVolumes
 - AttachVolume
 - DeleteVolume
 - ModifyInstanceAttribute
- To use the importKeyPairIfMissing property, Altus Director requires the following EC2 permission:
 - ImportKeyPair
- To use spot instances, the following additional EC2 permissions are required:
 - RequestSpotInstances
 - CancelSpotInstanceRequests
 - DescribeSpotInstanceRequests
- When working with encrypted EBS volumes (including AMIs with encrypted volumes) that use a custom key stored in KMS, Altus Director also requires the following KMS permissions:
 - DescribeKey
 - CreateGrant
 - ReEncrypt
 - GenerateDataKey
- To validate the templates used for EC2 instance creation, Altus Director requires permissions for the following IAM method:
 - GetInstanceProfile
- To create instances with [instance profiles](#), Altus Director requires permissions for the following IAM method:
 - PassRole
- To create RDS database servers for persistence on demand, Altus 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 Altus 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:GetConsoleOutput",
        "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",
        "kms:CreateGrant",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "directorIam",
      "Effect": "Allow",
      "Action": [
        "iam:GetInstanceProfile",
        "iam:PassRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "directorRds",
      "Effect": "Allow",
      "Action": [
        "rds:CreateDBInstance",
        "rds>DeleteDBInstance",
        "rds:DescribeDBInstances",
        "rds:DescribeDBEngineVersions",
        "rds:DescribeDBSubnetGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

```
} ]
```

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 Altus 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 Altus 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 Altus 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`.

Cloudera Manager Health Information

The following Cloudera Manager health information is available through Altus 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** - Altus 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 Altus Director server polls Cloudera Manager for health information. The default value is 30,000 ms (30 seconds). To disable health collection, set `lp.cache.health.pollingRateInMilliseconds` to 0.
- `lp.cache.health.numberOfWorkingCacheExecutorThreads` - The number of threads used to simultaneously request health information from Cloudera Manager. The default value is 5.
- `lp.cache.health.expirationMultiplier` - Used to determine if a health value is stale. If the health value has not been updated in `pollingRateInMilliseconds * expirationMultiplier` milliseconds, then the health value is considered stale and is reported to the web UI as `NOT_AVAILABLE`. Using the default settings, for

example, if health has not been reported in $2 * 30,000$ milliseconds = 60 seconds, it becomes stale. The default value is 2.



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

Opening Cloudera Manager

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

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

Altus 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.

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. Altus 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 Altus 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 Altus 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 Altus Director. These logs are not uploaded to Cloudera Support.

You can initiate diagnostic data collection manually through the Altus 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 Altus Director web UI or the Altus Director API.

Using the Web UI

To trigger diagnostic data collection, perform the following steps:

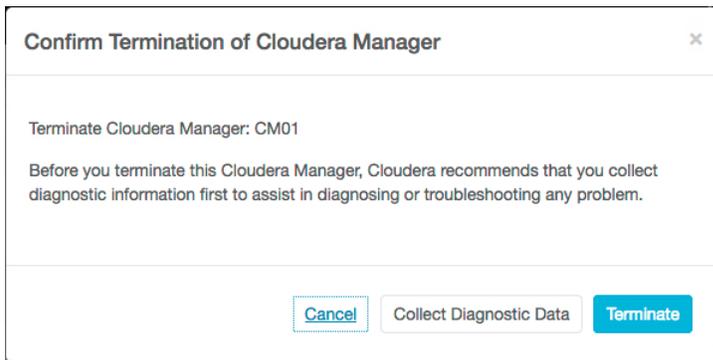
1. Go to the Altus Director web UI page for the deployment or cluster.
2. Click the down arrow on the dropdown list to the right of the deployment or cluster name.
3. In the dropdown list, click **Collect Diagnostic Data**.

Using Altus Director Server to Manage Cloudera Manager Instances

Altus 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 Altus Director `application.properties` file, also downloads a zip file containing the diagnostic data for the deployment or cluster to the Altus 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 Altus 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 Altus 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:

CM01 [Terminate Cloudera Manager](#)

Deployment Details

Status	Updating	Host	10.38.6.47:7180
Diagnostic Log Status	View Log Status	Instance	View Properties

Deployment Template

Instance Template	View Template	Configuration	Cloudera Manager Configurations
Image ID	ami-af4333cf	Provider Instance Type	m4.xlarge
License Type	Cloudera Enterprise	Kerberos	Not Enabled
Billing ID	*****k5epAAA	Repository	

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:

The screenshot displays the Cloudera Director interface for cluster CM01. At the top, there's a header with 'CM01' and an 'Add Cluster' button. Below this, the 'Deployment Details' section shows the cluster status as 'Ready', with fields for 'Host' (10.38.6.47:7180), 'URL' (Cloudera Manager), and 'Instance' (View Properties). A 'Diagnostic Log Status' link is visible, which is highlighted in the image. Below this, the 'Diagnostic Log Summary' is shown, containing the following information:

- Start Time: 6:53:52 AM PST
- Status: Done
- Diagnostic data was collected (with a green checkmark and a help icon)
- Diagnostic data was downloaded (with a green checkmark and a help icon)
- Cloudera Manager logs were not downloaded (with a red X and a help icon)
- File Path: /tmp/20170117145457_Env01_CM01_diagnostic-data_scom4.zip
- Note: This path can be found on your Cloudera Director server instance if Cloudera Director has been configured to download logs.

To the right of the summary, there's a 'Cloudera Manager Configurations' section showing 'm4.xlarge' and 'Not Enabled'. At the bottom right, there's a table for 'CDH version' with a value of '5' and a 'Modify Cluster' button.

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 Altus Director. You can configure diagnostic data collection on Cloudera Manager and Altus 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 Altus Director

Several Altus 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 Altus 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`.

Using Altus Director Server to Manage Cloudera Manager Instances

- `lp.debug.downloadDiagnosticData`: Determines whether diagnostic data is downloaded to the Altus Director instance. The default value is `true`.
- `lp.remote.terminate.assumeYes`: Determines whether Altus 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 Altus 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 Altus Director properties by using the CLI or editing the `application.properties` file, see [Setting Altus Director Properties](#) on page 171.

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 Altus 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 Altus 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 Altus Director actions.

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



Note: Cloudera Manager has many more user roles that control access to Cloudera Manager features. For more information see [Cloudera Manager User Roles](#) in the Enterprise documentation.

Managing Users with the Altus Director Web web UI

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

Create a User Account

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

1. On the Altus 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 Altus 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 Altus 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 Altus 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 Altus 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 Altus 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 Altus Director API

Altus 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 Altus 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.

REST method	Description
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 Altus Director API, see the server API documentation at *director-server-hostname:7189/api-console*. Expand the section labeled **users**.

Using Altus Director Server to Manage Cluster Instances

This section includes topics on managing CDH cluster instances:

The Altus Director Configuration File

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

For information on the 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 Altus 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 Altus Director, you can import the contents of a configuration file into Altus Director with the API using the `/api/<api_version>/import` endpoint. For example:

```
/api/v10/import?clusterName=someClusterName&deploymentName=someDeploymentName&environmentName=someEnvironmentName
```

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

Default Values for Configuration Files

In Altus Director 2.7 and lower, Altus Director provides default values for several common configuration file properties when no values have been specified in a user's configuration file.

In 2.8 and higher, Altus Director no longer provides the default values. You must include these values in any new configuration files you create for 2.8 and higher, and add them to older configuration files if you want to reuse them. Specifically, the following configuration file properties must now be included in your configuration file:

- **CDH version:** In Altus Director 2.8 and higher, the `products` section in the cluster description section of the configuration file must specify the CDH version, for example, `CDH: 5`, as shown below.

```
products {
  CDH: 5
  KAFKA: 3
  ACCUMULO: 1.6
  # SPARK2: 2
}
```

- **Instance name prefix:** In the absence of a user-provided instance name prefix, the default instance name prefix for AWS EC2 instances in Altus Director 2.7 and lower is `launchpad`, and for Microsoft Azure and Google Cloud Platform instances it is `director`. In 2.8 and higher, the default instance name prefix is `director` on all three supported cloud platforms. If you require the prefix `launchpad` for EC2 instances, you must specify it in your configuration file.
- **Instance template names based on EC2 instance types:** The instance template names `m1x`, `cc28`, and `hs18` can be specified in Altus Director 2.7 and lower to launch a particular EC2 instance type. If you want to use these as instance names in 2.8 and higher, you can specify them as names in your configuration file, but they won't automatically correspond to an instance type. You must configure the instance type yourself.
- **Predefined lists of master and worker roles for each cluster service:** No default master and worker role lists for CDH services are provided in Altus Director 2.8 and higher, and they must be specified in the configuration file. Existing configuration files that rely on the predefined lists of roles must be updated with master and worker roles if you want to use them with Altus Director 2.8 and higher.

Submitting a Cluster Configuration File

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

- Through the Altus Director server web UI.
- Through the Altus 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 Altus Director client to submit a configuration file. The configuration file will be applied to the cluster and managed by the Altus Director server.

When you submit a cluster configuration from a Altus Director client to the Altus Director server, all communications are transmitted in the clear (including the AWS credentials) unless you use TLS. If the client and server communicate over the Internet, Cloudera recommends that you either use TLS, or use a VPN for security. For information about configuring Altus Director to require TLS encryption for access, see [TLS for Altus Director Itself](#).



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 Altus Director server, follow these steps:

1. Create a configuration file. See [Provisioning a Cluster on AWS](#) on page 220.

2. Install the latest version of the Altus Director client from the [Altus Director Download Page](#).
3. Enter the following command:

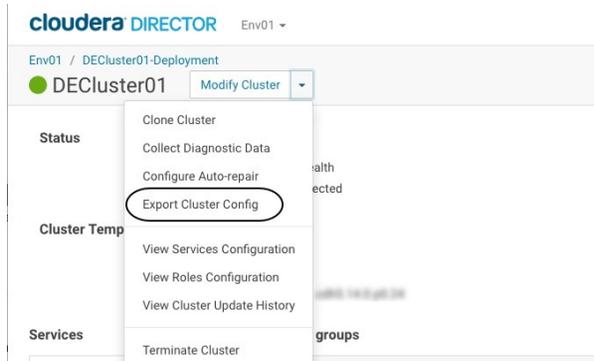
```
cloudera-director bootstrap-remote myconfig.conf --lp.remote.username=admin
--lp.remote.password=admin --lp.remote.hostAndPort=host:port
```

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 Altus Director server is running, and *port* is the port on which it is listening. The default port for Altus Director is 7189.

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

Exporting a Configuration File

You can export a client configuration file through the web UI or through the server API. To export a client configuration file through the web UI, go to the cluster page, click **Modify Cluster**, and select **Export Cluster Config**:



You can use the exported configuration file as is, or edit it as desired to create new clusters.



Note: For security reasons, some fields in the exported configuration file will be redacted. Apart from edits you may wish to make, values for these redacted fields must be provided in order to use the configuration file:

- Cloud provider credentials
- SSH private key
- Any database passwords

Deploying Clusters in an Existing Environment

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

1. Log in to Altus 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 Altus Director you are using:

Using Altus Director Server to Manage Cluster Instances

- If you are using Altus Director 2.0, the latest released version of Cloudera Manager/CDH 5.5 will be installed by default.
- If you are using Altus 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 <https://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 <https://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.

Altus 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.

Using Spot Instances

To help you manage your cloud resource costs, Cloudera supports AWS 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.

Altus 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 Altus Director 2.0 and higher, Cloudera recommends using Altus 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, Altus 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 205 below.)



Note: With versions of Altus Director lower than 2.5 and versions of Cloudera Manager and CDH lower 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 might 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.

Create New Instance Template

Root volume size (GB) ?

Root volume type ?

Use Spot Instances ? ←

Spot bid (USD/hr) ? ←

Spot Block Duration (minutes) ?

Tenancy ?

SSH username ?

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.

Create New Instance Template

Root volume size (GB) ?

Root volume type ?

Use Spot Instances ? ←

Spot bid (USD/hr) ? ←

Spot Block Duration (minutes) ? ←

Tenancy ?

SSH username ?

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.

Additional best practices for using spot instances can be found in [Spot Instance Interruptions](#) in the AWS documentation.

Using Products outside CDH with Altus Director

Products packaged in parcels separate from the CDH parcel can be included in clusters that are bootstrapped with Altus Director. Create a configuration file that includes the desired products (see [The Altus Director Configuration File](#) on page 199 for more information), and then use the Altus 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 Altus 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: [
    "https://archive.cloudera.com/spark2/csd/SPARK2_ON_YARN-2.0.0.cloudera2.jar",
    "https://archive.cloudera.com/kudu/csd/KUDU-5.10.0.jar",
  ]
}...
```

During deployment bootstrap, Altus 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 can 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, Altus Director installs CSD files into the custom location instead of the default location.

Using Altus Director Server to Manage Cluster Instances

CSD Support Before Altus Director 2.4

The ability to define CSDs for installation in deployment templates is a new feature for Altus Director 2.4. For earlier versions of Altus 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 can install additional CSDs into Cloudera Manager after Altus 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 can include the products corresponding to the CSDs, as if Altus Director itself had originally installed them.

Using Kudu with CDH 5.12 or Earlier

For Kudu 1.4 and earlier with CDH 5.12 and earlier, the Kudu service is distributed in its own parcel and is not part of CDH. To add Kudu to a CDH 5.12 or earlier cluster, bootstrapped by Altus 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 Altus 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: ["https://archive.cloudera.com/cdh5/parcels/5.11.0/",
                      "https://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
    }
  }
}
```


Using Altus Director Server to Manage Cluster Instances

```
}
cluster {
  products {
    CDH: 5.11.0,
    SPARK2: 2.3.0.cloudera2
  }
  parcelRepositories: ["https://archive.cloudera.com/cdh5/parcels/5.11.0/",
                      "https://archive.cloudera.com/spark2/parcels/2.3.0.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 Cloudera Data Science Workbench with Altus Director



Important: Cloudera Data Science Workbench is not currently supported with Cloudera Manager 6.0.0 and CDH 6.0.0. Consequently, you cannot use Altus Director 6 to deploy Cloudera Data Science Workbench on a C6 cluster. Cloudera Data Science Workbench will be supported with Cloudera Enterprise 6 in a future release.

Altus Director support for Cloudera Data Science Workbench is currently available for the following platforms and versions:

- **Amazon Web Services (AWS)** - Altus Director 2.6.0 (and higher)
- **Microsoft Azure** - Altus Director 2.7 (and higher)
- Cloudera Manager 5.13.1 (and higher 5.x versions)
- CSD-based Cloudera Data Science Workbench 1.2.x (and higher)

To create a configuration file to bootstrap a cluster that includes Cloudera Data Science Workbench, see the sample configuration file for your cloud provider:

- AWS: [aws.cdswh.conf](#)
- Microsoft Azure: [azure.cdswh.conf](#)

**Note:**

Users of Altus Director 2.7 and higher on AWS or Azure: The latest sample [aws.cds.conf](#) and [azure.cds.conf](#) files include an instance-level setting for `mountAllUnmountedDisks`. This must be set to **false**:

```
normalizationConfig {
  mountAllUnmountedDisks: false
}
```

Users of Altus Director 2.6 on AWS: Set the

`lp.normalization.mountAllUnmountedDisksRequired` global property to **false** in the Altus Director server's `application.properties` file and then restart Altus Director before you run the command to bootstrap a new cluster that includes Cloudera Data Science Workbench. (The `application.properties` file is located at `/etc/cloudera-director-server/application.properties` on the Altus Director server instance.)

A sample configuration file for clusters including Cloudera Data Science Workbench on Google Cloud Platform is not available at this time.

For more information about using Cloudera Data Science Workbench with Altus Director, see [Altus Director Support \(AWS and Azure Only\)](#) in the Cloudera Data Science Workbench documentation.

Using Third-Party Products with Altus 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 Altus Director.

Required Information

The following information is needed to successfully include a third-party product in an Altus 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 Altus Director cluster template. Parcels for different operating systems can 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 an Altus Director Configuration File

To add a third-party product to a cluster bootstrapped by Altus 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 Altus 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.

Using Altus Director Server to Manage Cluster Instances

Validation Warnings

Altus Director does not validate the services and roles that are part of a third-party product. It is normal for Altus 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 Altus Director will proceed with the bootstrap process. Check Cloudera Manager for details on any product installation or configuration errors.

Missing manifest.json File

Altus Director might 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.

Creating and Modifying Clusters with the Altus Director Web UI

Before initially launching a CDH cluster, you can use the Altus 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 Altus Director creates a CDH 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 Altus Director, a default set of instance groups and roles, based on the CDH services you include, is displayed in the Instance Groups section of the Add Cluster page:

Instance groups

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

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

Instance group: workers

×

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

Cancel Reset **OK**

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

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

Modifying the Number of Instances in an Existing Cluster

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

Using Altus Director Server to Manage Cluster Instances

How to Add Instances to a Cluster

1. Log in to Altus Director at `http://director-server-hostname:7189`. Altus 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.
4. Click **Continue** to confirm, and add the new instances or group to the cluster.



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 213.

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 Altus 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.

How to Remove Instances from a Cluster

1. Log in to Altus Director at `http://director-server-hostname:7189`.
Altus 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.
- Altus 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 213.

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. Altus 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 Altus Director at `http://director-server-hostname:7189`
Altus 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 [Migrating HDFS Master Roles](#) on page 161.

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. Altus 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 Altus Director API, but not through the Altus 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 Altus 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. Interpreters 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 might 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 Altus 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

```

Using Altus Director Server to Manage Cluster Instances

```
# in the order they are listed here.  Interpreters 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 Altus 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.  Interpreters 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 Altus Director. Use these variables in your scripts to communicate with Cloudera Manager and configure it after Altus 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. <div data-bbox="1094 873 1425 1167" style="border: 1px solid green; padding: 5px;">  Note: This variable is not available for deployment-level post-creation scripts, since Cloudera Manager deployments are not necessarily associated with a particular cluster. </div>
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.

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 Altus Director. For example, `http://cloudera_director_host:7189`.
Altus 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.

Using the Altus Director Client

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



Important: The Altus Director command line client's standalone functionality has been removed in Altus Director 6 and higher. The client is now used only for command line interactions with an Altus Director server. The following CLI commands have been removed:

- bootstrap
- status
- terminate
- update
- validate

The following CLI commands are available for use with the Altus Director server:

- bootstrap-remote
- terminate-remote
- validate-remote
- convert-remote

Installing Altus Director Client

To install Altus Director client without Altus Director server, perform the tasks below. You must be either running as root or using sudo to perform these tasks.

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

- For AWS, see [Installing Altus Director Server and Client on the EC2 Instance](#) on page 34.
- For Google Cloud Platform, see [Installing Altus Director Server and Client on Google Compute Engine](#) on page 53.



Important: Altus Director requires a JDK. For more information, see [Supported Software and Distributions](#) on page 24.

1. Install a supported version of the Oracle Java Development Kit (JDK) on the Altus Director host. Altus Director 6.0 supports JDK version 8.



Note: Spark 2.2 and higher requires JDK 8 and Python 2.7 or higher.

For installation information, see [Java SE Downloads](#).

2. Download Altus Director by running the correct commands for your distribution.

- For RHEL 6 and CentOS 6:

```
cd /etc/yum.repos.d/
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat6/cloudera-director.repo"
```

Using the Altus Director Client

- For RHEL 7 and CentOS 7:

```
cd /etc/yum.repos.d/  
sudo wget "http://username:password@archive.cloudera.com/p/director6/6.0/redhat7/cloudera-director.repo"
```

- For Ubuntu 14.04:

```
cd /etc/apt/sources.list.d  
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1404/cloudera-director.list"  
-O
```

- For Ubuntu 16.04 :

```
cd /etc/apt/sources.list.d  
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1604/cloudera-director.list"  
-O
```

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, run the following command:

```
sudo curl -s -L "https://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1404/archive.key" | sudo  
apt-key add -
```

- For Ubuntu 16.04, run the following command:

```
sudo curl -s -L "https://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1604/archive.key" | sudo  
apt-key add -
```

4. Install Altus 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 or 16.04:

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

Provisioning a Cluster on AWS

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

You can use the `validate-remote` CLI command to check your configuration file's settings for environments, deployments, and clusters. For more information, see the entry for `validate-remote` on the page [Altus 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 Altus Director Client

After you modify the configuration file, you can run Altus Director client.

The following CLI commands are available for use with the Altus Director server:

- bootstrap-remote
- terminate-remote
- validate-remote
- convert-remote

For more information on using the client to deploy clusters on the server, see [Submitting a Cluster Configuration File](#).

1. From the cluster launcher, enter the following:

```
[ec2-user@ip-10-1-1-18]$ cloudera-director bootstrap-remote cluster.conf
--lp.remote.username=admin --lp.remote.password=admin
```

The default administrative user name and password for Altus Director is *admin*. If you change the administrative user name and password, run the bootstrap-remote command with the new administrative user name and password.



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 Altus 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.

Altus Director Upgrade

By default, Altus Director 6.0 creates clusters with Cloudera Manager 6.0 and CDH 6.0. When you upgrade to Director 6.0, you can continue to manage existing clusters with Cloudera Manager and CDH parcels 5.7 and higher.

Altus Director 6.0 has a number of changes that are not compatible with Altus Director 2.8. Before you start the upgrade process, review the list of changes in the release notes to understand the effects of upgrading to Altus Director 6.0. See [Changes in Altus Director 6.0](#) on page 16.

To upgrade to Altus Director 6.0, complete the following steps:

1. Review the upgrade requirements.

Altus Director 6.0 has different requirements than the previous version. Make sure that you understand the requirements before you start the upgrade process.

2. Perform the tasks required for the upgrade.

If your current Altus Director does not meet the Altus Director 6.0 requirements, the upgrade process will fail. For example, you cannot upgrade from an Altus Director version that is earlier than version 2.8. You must upgrade to Altus Director 2.8 before you start the upgrade to Altus Director 6.0.

3. Upgrade to Altus Director 6.0.

Follow the instructions for upgrading the Altus Director server and client on your operating system.

4. Restore the settings to match the configuration of the previous version.

If you customized the configuration of the Altus Director 2.8 instance, you can restore the configuration from your backup files and use the same configuration in Altus Director 6.0. Note that the names of some configuration properties have changed in Altus Director 6.0. Review the changes to verify that you are using the correct property names in Altus Director 6.0.

Complete this step before you start the Altus Director 6.0 server.

5. Restart the Altus Director 6.0 server.

Step 1: Review the Upgrade Requirements

Altus Director 6.0 has the following requirements:

Only Altus Director 2.8 can be upgraded to Altus Director 6.0.

You cannot upgrade to Altus Director 6.0 from any other Director version.

Altus Director 6.0 requires Oracle JDK version 8.

Altus Director 6.0 supports only Java 8. Altus Director 6.0 does not support earlier or later versions of Java.

Altus Director 6.0 requires Cloudera Manager version 5.7 or higher.

Altus Director 6.0 does not work with Cloudera Manager instances earlier than version 5.7.

Altus Director 6.0 supports RHEL and CentOS versions 6.7, 6.8, 7.2, 7.4, and 7.5 for Cloudera Manager and clusters.

Altus Director 6.0 supports installation of Cloudera Manager and clusters on the following OS versions:

- For AWS and Google Cloud Platform only, Altus Director 6.0 now supports RHEL and CentOS version 7.5
- For all cloud service providers, Altus Director 6.0 continues to support RHEL and CentOS versions 6.7, 6.8, 7.2, and 7.4

Altus Director 6.0 cannot create clusters on other versions of the operating systems. Likewise, Altus Director 6.0 does not work with existing Cloudera Manager and clusters that are running on other versions of the operating systems.

For the complete list of requirements for Altus Director, see [Requirements and Supported Versions](#).

Step 2. Perform Tasks Required for the Upgrade

Complete all the necessary tasks to meet the upgrade requirements to ensure that you can successfully upgrade to Altus Director 6.0.

Before you upgrade to Altus Director 6.0, complete the following tasks:

Verify that you are upgrading from Director 2.8 to Altus Director 6.0.

Required.

You can upgrade to Altus Director 6.0 only from Altus Director 2.8. If you are upgrading from an Altus Director version that is earlier than version 2.8, you must upgrade to version 2.8 before you can upgrade to Altus Director 6.0.

The Altus Director 6.0 upgrade will not start if it detects that the Altus Director instance you are upgrading is not version 2.8.

For more information about upgrading to Altus Director 2.8, see the [Altus Director 2.8 upgrade documentation](#).

Verify that your Java version is Oracle Java 8.

Required.

Altus Director 6.0 requires Oracle JDK version 8. It does not support other versions of the Oracle JDK. You must upgrade to Java 8 before you can upgrade to Altus Director 6.0.

For more information about Java installations, see the [Oracle Java page](#).

Upgrade all Cloudera Manager instances that are older than version 5.7

Required.

Altus Director 6.0 supports Cloudera Manager version 5.7 or higher. Verify that all Director deployments are running Cloudera Manager version 5.7 or higher and upgrade any Cloudera Manager version 5.6 or earlier.

Altus Director 6.0 will not start if it detects a deployment running Cloudera Manager 5.6 or earlier.

Back up the Altus Director 2.8 properties file.

Recommended.

If you have modified the Director 2.8 *application.properties* file, back up the file before you start the upgrade.

The *application.properties* is located in the following directory: `/etc/cloudera-director-server/`

After the upgrade, you can use the settings in the backup file to set the same configuration in Altus Director 6.0.

Back up the configuration files for the Altus Director 2.8 plug-ins.

Recommended.

Altus Director includes configuration files that enable you to configure how the Altus Director plug-ins work. If you have modified the Altus Director 2.8 plug-in configuration files, back up the configuration files before you start the upgrade. After the upgrade, you can copy the plug-in configuration files from your backup and use the same configuration files for the Altus Director 6.0 plug-ins.

The Altus Director plug-in files are in the following directory: `/var/lib/cloudera-director-plugins/`

The directory contains a separate versioned subdirectory for each plug-in. Back up the directory for the plug-in where you modified the configuration file. Or, if you modified the configuration files for multiple plug-ins, back up the main `/cloudera-director-plugins` directory.

Back up the Altus Director 2.8 database that stores state information.

Recommended.

The Altus Director 6.0 upgrade overwrites the state information in the Altus Director database. Back up the database before you start the upgrade process. If the upgrade fails and you decide to revert back to the previous version, you can restore the data.

By default, Altus Director stores state information in the embedded H2 database named *state.h2.db* in the following directory: `/var/lib/cloudera-director-server/`. If you are using the default database, back up the *state.h2.db* file.

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

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

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

Step 3. Upgrade to Altus Director 6.0

The upgrade procedure differs based on the operating system where the current Altus Director is installed.



Note: Allow all running operations to complete before you start the upgrade.

RHEL and CentOS

1. Stop the Altus Director server service by running the following command:

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

Before you proceed, verify that the Altus Director instance that you are upgrading meets all upgrade requirements. See [Step 1: Review the Upgrade Requirements](#) on page 223.

2. Download the Altus Director `cloudera-director.repo` file for Altus Director 6.0.

The `cloudera-director.repo` file is the YUM repository configuration file for Altus Director. The base URL in the latest `cloudera-director.repo` file points to the Altus Director 6.0 repository.

Run the following commands to download the latest Altus Director `cloudera-director.repo` file :

RHEL and CentOS 7.x:

```
cd /etc/yum/repos.d/
sudo wget "https://username:password@archive.cloudera.com/p/director6/6.0.0/redhat7/cloudera-director.repo"
```

RHEL and CentOS 6.x:

```
cd /etc/yum/repos.d/
sudo wget "https://username:password@archive.cloudera.com/p/director6/6.0.0/redhat6/cloudera-director.repo"
```

3. Run the following commands to upgrade the Altus Director server and client:

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

The Altus Director upgrade installs all required plug-in packages.

After the Altus Director server and client upgrade completes, rerun the service manager in the operating system.

4. Run the following command to re-execute the `systemd` manager:

```
sudo systemctl daemon-reexec
```



Note: Before you restart the server, restore the configuration from the previous version if you want to use the same configuration in Altus Director 6.0. See [Step 4. Restore the Configuration](#) on page 226.

Ubuntu

1. Stop the Altus Director server service by running the following command:

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

Before you proceed, verify that the Altus Director instance that you are upgrading meets all upgrade requirements. See [Step 1: Review the Upgrade Requirements](#) on page 223.

2. Download the Altus Director *cloudera-director.list* file for Altus Director 6.0.

The *cloudera-director.list* file is the repository configuration file for Altus Director. The base URL in the latest *cloudera-director.list* file points to the Altus Director 6.0 repository.

Run the following commands to download the latest Altus Director *cloudera-director.list* file :

Ubuntu 14.04

```
cd /etc/apt/sources.list.d/
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1404/cloudera-director.list"
-o
```

Ubuntu 16.04:

```
cd /etc/apt/sources.list.d/
sudo curl -L "http://username:password@archive.cloudera.com/p/director6/6.0/ubuntu1604/cloudera-director.list"
-o
```

3. Run the following commands to upgrade the Altus Director server and client:

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

The Altus Director upgrade installs all required plug-in packages.



Note: Before you restart the server, restore the configuration from the previous version if you want to use the same configuration in Altus Director 6.0. See [Step 4. Restore the Configuration](#) on page 226.

Step 4. Restore the Configuration

If you want Altus Director 6.0 to have the same configuration as the previous version, restore the configuration settings from the backup files. Do not restart the Altus Director 6.0 server until you have updated the configuration files.

Complete the following tasks:

Restore the Altus Director configuration.

If you modified the *application.properties* file in the previous version, you can use the same configuration in Altus Director 6.0. Restore the configuration from the backup file.

In Altus Director 6.0, the names of the following properties in the *application.properties* have changed:

Old property name	New property name
lp.terminate.deployment.clouderaManagerServerStopWaitTimeInMs	lp.cleanup.cm.stopWaitTimeInMs
lp.terminate.deployment.timeBetweenConnectionRetriesInMs	lp.cleanup.cm.timeBetweenConnectionRetriesInMs

Before you overwrite the *application.properties* file in Altus Director 6.0, check whether you have modified any of the properties that have new names. Update the *application.properties* backup file before you copy it to the `/etc/cloudera-director-server/` directory in Altus Director 6.0.

Restore the plug-in configuration files.

If you modified the configuration files for any of the Altus Director plug-ins in the previous version, restore the configuration files to the Altus Director 6.0 plug-in directory before you restart the Altus Director 6.0 server.

The Altus Director plug-in files are in the following directory: `/var/lib/cloudera-director-plugins/`

The directory contains a separate versioned subdirectory for each plug-in. Although the version suffix in the subdirectory names changes for every version, the plug-in directory structure and plug-in configuration file names in Altus Director 6.0 is the same as in previous versions.

You can copy the plug-in configuration files from your backup and overwrite the corresponding configuration files in the Altus Director 6.0 plug-in directory.

Step 5. Restart Director

To start Altus Director 6.0, run the following command:

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

The command to start Altus Director is the same in RHEL, CentOS, and Ubuntu.

Troubleshooting Altus Director

This topic contains information on problems that can occur when you set up, configure, or use Altus Director, their causes, and their solutions.

Viewing Altus Director Logs

To help you troubleshoot problems, you can view the Altus Director logs. Log files are in the following locations:

- Altus Director client
 - One shared log file per user account:

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

- Altus Director server
 - One file for all clusters:

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

Altus 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:

- Altus Director client: `/etc/cloudera-director-client/logback.xml`
- Altus Director server: `/etc/cloudera-director-server/logback.xml`

The XML file configures the logback logging library. To turn on all debug logging for Altus Director and its libraries, change the "root" element as follows:

```
<root level="DEBUG">
```

Enabling debug logging significantly increases the size of the logs, and can 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 Altus 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.

Configuring Tag-on-create for AWS GovCloud (US) and China (Beijing) Regions

In most AWS regions, Altus Director assigns a tag during the creation of each instance it creates to facilitate instance management. The GovCloud (US) and China (Beijing) regions do not support tagging of instances on creation, so for instances in these regions, the tag is created after the instance is created.

If you are running Altus Director in the GovCloud (US) or China (Beijing) regions, you must turn off `useTagOnCreate` in the Altus Director AWS plugin. To do this add the following line to the end of the `aws-plugin.conf` file, after the last closing bracket:

```
useTagOnCreate: false
```

The `aws-plugin.conf` file can be found at

`/var/lib/cloudera-director-plugins/aws-provider-plugin_version/etc/` on your Altus Director EC2 instance.

Backing Up the H2 Embedded Database

By default, Altus 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 Altus Director Database

Manual modifications to the Altus Director database *are not supported*. Modifications made outside of Altus Director control can lead to permanent data loss and unrecoverable errors in Altus Director.

Bootstrap fails in Azure when custom image has an attached data disk and dataDiskCount is not 0

Symptom

Bootstrap fails in Azure when a custom image is used that has an attached data disk and **dataDiskCount** is not set to 0. The error message displayed is, "Cannot specify user image overrides for a disk already defined in the specified image reference."

Cause

This error originates in Azure. It occurs because the Azure image has a data disk attached, while the **dataDiskCount** value wrongly indicates that Altus Director is trying to attach an additional disk or disks. The conflict causes an error to be thrown.

Solution

If you deploy a cluster in Azure with a custom image that has a data disk attached, you must set **dataDiskCount** to 0. You can use the **Azure Portal** to check if your custom image has a data disk attached. If you simply comment out the **dataDiskCount** setting, it will default to 5. Bootstrap fails if the **dataDiskCount** value is not 0. See [Deploying Clusters with Custom Images](#) on page 87.

Slow or Failed OS Updates in Some AWS Regions

Symptom

In AWS, Altus 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 might go slowly or fail.

Solutions

Consider trying one or more of the following steps:

- **Disable instance normalization.** This causes Altus 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.

- **Create a preloaded AMI.** Altus 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 Altus 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.

Altus Director Bootstrap Fails with DNS Error

Symptom

Altus Director fails to bootstrap with the error message, "DNS is not configured correctly on at least one instance."

Cause

Needs to be diagnosed.

Solution

Verify that DNS is configured properly. Check the server logs, which might contain additional warning messages and information about why DNS detection failed. For example, this error can appear when an invalid ssh user has been set.

Altus Director Bootstrap Fails with IAM Permissions Error

Symptom

Altus Director fails to bootstrap with the error message:

```
ErrorInfo{code=PROVIDER_EXCEPTION, properties={message=User:
arn:aws:sts::code:assumed-role/ClouderaDirector-Director-instance
is not authorized to perform: iam:GetInstanceProfile on resource: instance profile test}}
```

Cause

User failed to configure the required IAM permissions.

Solution

Configure the required IAM permissions. Check the list of required IAM permissions: [Creating AWS Identity and Access Management \(IAM\) Policies](#) on page 189.

Cloudera Manager API Call Fails

Symptom

A Cloudera Manager API call fails in Altus 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.

Altus Director Cannot Manage a Cluster That Was Kerberized Through Cloudera Manager

Symptom

Altus Director cannot manage a cluster after Cloudera Manager is used to enable Kerberos on the cluster.

Cause

Once a cluster is deployed through Altus Director, some changes to the cluster that are made using Cloudera Manager cause Altus Director to be out of sync and unable to manage the cluster. See [Altus Director and Cloudera Manager Usage](#) on page 179.

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

Altus Director does not validate that all required roles are assigned when provisioning a cluster. This can lead to failures during the initial run of a new cluster. For example, if the gateway instance group was removed, but the Flume Agent and Kafka Broker were assigned to roles in that group, the cluster 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.

Altus Director Server Will Not Start with Unsupported Java Version

Symptom

Altus 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 Altus Director server against an older, unsupported version of the Oracle Java SE Development Kit (JDK).

Solution

Update to a supported version of the Oracle JDK. For information on supported versions, see [Supported Software and Distributions](#) on page 24.

Error Occurs if Tags Contain Unquoted Special Characters

Symptom

When using the `bootstrap-remote` command to set up a cluster with Altus 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 might 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

Altus 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

Altus 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/store URL:  
http://ip-10-202-202-109.ec2.internal:7180/cmF/parcel/download/CDH-5.10.0-1.cdh5.10.0.p0.41-e17.parcel.torrent
```

```
->
/opt/cloudera/parcel-cache/CDH-5.10.0-1.cdh5.10.0.p0.41-e17.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 Altus 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 Altus Director server does not start or quickly exits with an Out of Memory exception.

Troubleshooting Altus Director

Cause

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

Solution

Run Altus Director on an instance that has at least 1 GB of free memory. See [Resource Requirements](#) on page 26 for more details on Altus 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 [How to Remove Instances from a Cluster](#) on page 212 (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 Altus Director Server

Symptom

You are unable to connect to the Altus Director server.

Cause

Configuration of security group and iptables settings. For more information about configuring security groups, see [Preparing Your AWS EC2 Resources](#) on page 29. For commands to turn off iptables, see either [Installing Altus Director Server and Client on the EC2 Instance](#) on page 34 or [Installing Altus Director Server and Client on Google Compute Engine](#) on page 53. 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 Altus Director.

General Questions

Can I move Altus Director to a different instance?

In some circumstances, you might want to move Altus Director from the instance where it is installed to a different instance. This can be done much as it would be with any other application. One straightforward procedure is to snapshot or image the current instance and deploy a new instance from it.

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. If you are using MySQL to host the Altus Director database, then the database resides elsewhere on a database server. Backing up the Altus Director database in either case is an extra precaution, but isn't strictly necessary in order to move Altus Director over to the new instance.



Note: If you are using MySQL for the Altus Director database and the new Altus Director instance is in a new subnet, the security rules need to allow instances in the new subnet to reach the MySQL server, whether in a provider database service 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 192.

How can I make Altus Director highly available?

Altus 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 Altus 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 Altus 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, Altus Director has the ability to resume interrupted work.

For information on setting up highly available clusters in a Cloudera Manager deployment using Altus Director, see [Creating Highly Available Clusters With Altus Director](#) on page 158.

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
```

Frequently Asked Questions

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.

Altus 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.

Altus 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 Altus Director and the configuration file.

configuration file

A template file used by Altus 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 it easier to find instances using environment management tools. For example, AWS provides the AWS Management Console.

template

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

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.
```