



Cloudera Enterprise Upgrade Guide

Important Notice

© 2010-2019 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 Enterprise Upgrade Guide Cloudera Enterprise Upgrade Guide
Date: April 11, 2019

Table of Contents

Cloudera Enterprise Upgrade Guide.....10

Upgrade Overview.....	10
Assessing the Impact of an Upgrade.....	11
Overview of Upgrading Cloudera Manager.....	12
Overview of Upgrading CDH.....	12
Overview of Upgrading Cloudera Navigator Components.....	13

Install and Upgrade Notes.....14

Upgrades to Cloudera Enterprise 6.x.....	14
Restart of Impala and Hive required for Cloudera Manager 6.2 upgrade with ADLS.....	14
TSB-359 Backup and Disaster Recovery (BDR) HDFS and Hive Replications will fail on clusters running Cloudera Manager 6.1.0.....	14
Upgrades from Cloudera Enterprise 5.15 or 5.16 to 6.0x are not supported.....	15
Upgrading to CDH 6.1.0 Enables Direct SQL mode in Hive service by default.....	15
Upgrades from Cloudera Enterprise 6.0 Beta Release to 6.x General Release Not supported	15
Cloudera Express License Enforcement.....	15
Cloudera Data Science Workbench is Not Supported with Cloudera Enterprise 6.0.....	15
Impala roles with SELECT or INSERT privileges receive REFRESH privileges during the upgrade	15
Hue requires manual installation of psycopg2.....	15
CDH Upgrade fails to delete Solr data from HDFS.....	16
Package Installation of CDH Fails.....	16
Uninstall CDH 5 Sqoop connectors for Teradata and Netezza before upgrading to CDH 6.....	16
Unsupported Sqoop options cause upgrade failures.....	16
Generated Avro code from CDH 5 should be regenerated when upgrading.....	17
Upgrading Apache Parquet to CDH 6.....	17
No HBase Replication Peer Configuration Change During Rolling Update.....	17
Oracle Database Initialization.....	17
TLS Protocol Error with OpenJDK.....	17
Upgrades to Cloudera Enterprise 5.x.....	18
Flume Kafka client incompatible changes in CDH 5.8.....	18
Upgrade to CDH 5.13 or higher Requires Pre-installation of Spark 2.1 or Spark 2.2.....	18
Sentry may require increased Java heap settings before upgrading CDH to 5.13.....	18
Apache MapReduce Jobs May Fail During Rolling Upgrade to CDH 5.11.0 or CDH 5.11.1.....	18
Cloudera Manager set catalogd default jvm memory to 4G can cause out of memory error on upgrade to Cloudera Manager 5.7 or higher.....	19

Supported Upgrade Paths20

Supported CDH Upgrade Paths.....	20
Cloudera Manager support for CDH.....	20

Upgrading the JDK.....22

Manually Installing Oracle JDK 1.8.....	23
OpenJDK.....	24
Manually Installing OpenJDK.....	24
Manually Migrating from Oracle JDK to OpenJDK.....	25
Using AES-256 Encryption.....	27
Configuring a Custom Java Home Location.....	28

Upgrading the Operating System.....29

Getting Started with Operating System Upgrades.....	29
Prerequisites.....	29
Backing Up Host Files Before Upgrading the Operating System.....	29
Backing Up.....	29
Before You Upgrade the Operating System.....	30
Decommission and Stop Running Roles.....	30
Stop Cloudera Manager Agent.....	31
Stop Cloudera Manager Server & Agent.....	31
Stop Databases.....	32
Remove Packages & Parcels.....	32
Upgrade the Operating System.....	33
After You Upgrade the Operating System.....	33
Establish Access to the Software.....	33
Reinstall Cloudera Manager Daemon & Agent Packages.....	35
Reinstall Cloudera Manager Server, Daemon & Agent Packages.....	35
Start Databases.....	36
Start Cloudera Manager Server & Agent.....	36
Start Roles.....	37

Upgrading Cloudera Manager.....38

Getting Started Upgrading Cloudera Manager.....	38
.....	39
Collect Information.....	40
Preparing to Upgrade Cloudera Manager.....	40
Backing Up Cloudera Manager.....	41
Collect Information for Backing Up Cloudera Manager.....	41
Back Up Cloudera Manager Agent.....	43
Back Up the Cloudera Management Service.....	43
Back Up Cloudera Navigator Data.....	44

<i>Stop Cloudera Manager Server & Cloudera Management Service.....</i>	<i>44</i>
<i>Back Up the Cloudera Manager Databases.....</i>	<i>45</i>
<i>Back Up Cloudera Manager Server.....</i>	<i>46</i>
<i>(Optional) Start Cloudera Manager Server & Cloudera Management Service.....</i>	<i>46</i>
<i>Upgrading the Cloudera Manager Server.....</i>	<i>47</i>
<i>Establish Access to the Software.....</i>	<i>48</i>
<i>Install JDK 8.....</i>	<i>49</i>
<i>Upgrade the Cloudera Manager Server.....</i>	<i>50</i>
<i>Upgrading the Cloudera Manager Agents.....</i>	<i>54</i>
<i>Upgrade the Cloudera Manager Agents.....</i>	<i>55</i>
<i>After You Upgrade Cloudera Manager.....</i>	<i>60</i>
<i>Upgrade Cloudera Navigator Encryption Components.....</i>	<i>61</i>
<i>Perform Post Upgrade Steps.....</i>	<i>61</i>
<i>Troubleshooting a Cloudera Manager Upgrade.....</i>	<i>62</i>
<i>The Cloudera Manager Server fails to start after upgrade.....</i>	<i>62</i>
<i>Re-Running the Cloudera Manager Upgrade Wizard.....</i>	<i>62</i>
<i>TLS Protocol Error with OpenJDK.....</i>	<i>63</i>
<i>Reverting a Failed Cloudera Manager Upgrade.....</i>	<i>63</i>
<i>Ensure Cloudera Manager Server and Agent are stopped.....</i>	<i>64</i>
<i>Restore the Cloudera Manager Database (if necessary).....</i>	<i>64</i>
<i>Establish Access to the Software.....</i>	<i>65</i>
<i>Downgrade the Cloudera Manager Packages.....</i>	<i>66</i>
<i>Restore the Cloudera Manager Directory.....</i>	<i>67</i>
<i>Start Cloudera Manager Again.....</i>	<i>68</i>

Upgrading CDH.....69

<i>Getting Started Upgrading CDH.....</i>	<i>69</i>
<i>.....</i>	<i>70</i>
<i>Collect Information.....</i>	<i>70</i>
<i>Preparing to Upgrade CDH.....</i>	<i>71</i>
<i>Backing Up CDH.....</i>	<i>73</i>
<i>Back Up HDFS Metadata on the NameNode.....</i>	<i>73</i>
<i>Back Up the Repository Files.....</i>	<i>74</i>
<i>Back Up Databases.....</i>	<i>74</i>
<i>Back Up ZooKeeper.....</i>	<i>75</i>
<i>Back Up HDFS</i>	<i>75</i>
<i>Back Up Key Trustee Server and Clients.....</i>	<i>77</i>
<i>Back Up HSM KMS.....</i>	<i>77</i>
<i>Back Up Navigator Encrypt.....</i>	<i>77</i>
<i>Back Up HBase.....</i>	<i>77</i>
<i>Back Up Search.....</i>	<i>77</i>
<i>Back Up Sqoop 2.....</i>	<i>78</i>
<i>Back Up Hue.....</i>	<i>78</i>

CDH 6 Pre-Upgrade Migration Steps.....	78
<i>Migrating from Sentry Policy Files to the Sentry Service.....</i>	<i>78</i>
<i>Migrating Cloudera Search Configuration Before Upgrading to CDH 6.....</i>	<i>79</i>
<i>Migrating Apache Spark Before Upgrading to CDH 6.....</i>	<i>86</i>
<i>Migrating Apache HBase Before Upgrading to CDH 6.....</i>	<i>87</i>
<i>Installing Dependencies for Hue.....</i>	<i>89</i>
<i>Pre-Upgrade Migration Steps for Upgrading Key Trustee KMS to CDH 6.....</i>	<i>91</i>
<i>Pre-Upgrade Migration Steps for Upgrading HSM KMS to CDH 6.....</i>	<i>91</i>
Upgrading the CDH Cluster.....	92
<i>Back Up Cloudera Manager.....</i>	<i>94</i>
<i>Enter Maintenance Mode.....</i>	<i>94</i>
<i>Complete Pre-Upgrade Migration Steps</i>	<i>94</i>
<i>Establish Access to the Software.....</i>	<i>96</i>
<i>Run Hue Document Cleanup.....</i>	<i>97</i>
<i>Check Oracle Database Initialization.....</i>	<i>98</i>
<i>Stop the Cluster.....</i>	<i>98</i>
<i>Install CDH Packages.....</i>	<i>98</i>
<i>Download and Distribute Parcels.....</i>	<i>100</i>
<i>Run the Upgrade CDH Wizard.....</i>	<i>101</i>
<i>Remove the Previous CDH Version Packages and Refresh Symlinks.....</i>	<i>103</i>
<i>Complete the Cloudera Search Upgrade.....</i>	<i>104</i>
<i>Finalize the HDFS Upgrade.....</i>	<i>104</i>
<i>Finalize the HDFS Upgrade.....</i>	<i>105</i>
<i>For Sentry with an Oracle Database, Add the AUTHZ_PATH.AUTHZ_OBJ_ID Index.....</i>	<i>105</i>
<i>Complete Post-Upgrade Migration Steps.....</i>	<i>105</i>
<i>Exit Maintenance Mode.....</i>	<i>106</i>
CDH 6 Post-Upgrade Migration Steps.....	106
<i>Impala Upgrade Considerations.....</i>	<i>106</i>
<i>Re-Indexing Solr Collections After Upgrading to CDH 6.....</i>	<i>110</i>
<i>Apache Spark Post Upgrade Migration Steps.....</i>	<i>110</i>
<i>Migrating from MapReduce 1 (MRv1) to MapReduce 2 (MRv2).....</i>	<i>111</i>
<i>Upgrade Notes for Kudu 1.9 / CDH 6.2.0.....</i>	<i>123</i>
Upgrading CDH Manually after an Upgrade Failure.....	124
<i>Start ZooKeeper.....</i>	<i>124</i>
<i>Start Kudu.....</i>	<i>124</i>
<i>Upgrade HDFS Metadata.....</i>	<i>124</i>
<i>Start HDFS.....</i>	<i>125</i>
<i>Start HBASE.....</i>	<i>125</i>
<i>Upgrade the Sentry Database.....</i>	<i>125</i>
<i>Start Sentry.....</i>	<i>125</i>
<i>Start KAFKA.....</i>	<i>125</i>
<i>Upgrade Solr.....</i>	<i>125</i>
<i>Start FLUME.....</i>	<i>126</i>
<i>Upgrade KeyStore Indexer.....</i>	<i>126</i>

<i>Start Key-Value Store Indexer.....</i>	<i>126</i>
<i>Upgrade YARN.....</i>	<i>126</i>
<i>Install MR Framework Jars.....</i>	<i>127</i>
<i>Start YARN.....</i>	<i>127</i>
<i>Deploy Client Configuration Files.....</i>	<i>127</i>
<i>Upgrade the Spark Standalone Service.....</i>	<i>127</i>
<i>Start Spark Services.....</i>	<i>127</i>
<i>Upgrade the Hive Metastore Database.....</i>	<i>127</i>
<i>Start Hive.....</i>	<i>128</i>
<i>Validate the Hive Metastore Database Schema.....</i>	<i>128</i>
<i>Start Impala.....</i>	<i>128</i>
<i>Upgrade Oozie.....</i>	<i>128</i>
<i>Upgrade the Oozie SharedLib.....</i>	<i>128</i>
<i>Start Remaining Cluster Services.....</i>	<i>128</i>
<i>Test the Cluster and Finalize HDFS Metadata.....</i>	<i>129</i>
<i>Troubleshooting CDH Upgrades.....</i>	<i>129</i>
<i>"Access denied" in install or update wizard.....</i>	<i>129</i>
<i>Cluster hosts do not appear.....</i>	<i>130</i>
<i>Cannot start services after upgrade.....</i>	<i>130</i>
<i>HDFS DataNodes fail to start.....</i>	<i>130</i>
<i>Cloudera services fail to start.....</i>	<i>130</i>
<i>Upgrading to CDH 5.8.0 or CDH 5.8.1 When Using the Flume Kafka Client.....</i>	<i>131</i>
<i>Rolling Back a CDH 5 to CDH 6 Upgrade.....</i>	<i>132</i>
<i>Review Limitations.....</i>	<i>133</i>
<i>Stop the Cluster.....</i>	<i>133</i>
<i>(Parcels) Downgrade the Software.....</i>	<i>133</i>
<i>Stop Cloudera Manager</i>	<i>134</i>
<i>(Packages) Downgrade the Software.....</i>	<i>134</i>
<i>Restore Cloudera Manager Databases.....</i>	<i>136</i>
<i>Restore Cloudera Manager Server.....</i>	<i>136</i>
<i>Start Cloudera Manager</i>	<i>137</i>
<i>Roll Back ZooKeeper.....</i>	<i>138</i>
<i>Roll Back HDFS.....</i>	<i>138</i>
<i>Start the Key Management Server.....</i>	<i>141</i>
<i>Start the HBase Service.....</i>	<i>141</i>
<i>Restore CDH Databases.....</i>	<i>141</i>
<i>Start the Sentry Service.....</i>	<i>142</i>
<i>Roll Back Cloudera Search.....</i>	<i>142</i>
<i>Roll Back Hue.....</i>	<i>143</i>
<i>Roll Back Kafka.....</i>	<i>143</i>
<i>Roll Back Sqoop 2.....</i>	<i>143</i>
<i>Deploy the Client Configuration.....</i>	<i>143</i>
<i>Restart the Cluster.....</i>	<i>144</i>
<i>Roll Back Cloudera Navigator Encryption Components.....</i>	<i>144</i>
<i>(Optional) Cloudera Manager Rollback Steps.....</i>	<i>147</i>

Upgrading Cloudera Navigator Data Encryption.....152

Upgrading Cloudera Navigator Key Trustee Server.....	152
Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher.....	152
Upgrading Cloudera Navigator Key Trustee Server 3.x to 5.4.x.....	157
Upgrading Cloudera Navigator Key Trustee Server 3.8 to 5.5 Using the ktupgrade Script.....	163
Upgrading Cloudera Navigator Key HSM.....	167
Setting Up an Internal Repository.....	167
Upgrading Key HSM (Minor and Patch Version Upgrades).....	167
Upgrading Key HSM (Major Version Upgrades).....	168
Upgrading Key Trustee KMS.....	170
Setting Up an Internal Repository.....	170
Validating Private Key Synchronization (Key Trustee KMS HA Only).....	170
Upgrading Key Trustee KMS Using Parcels.....	171
Upgrading Key Trustee KMS Using Packages.....	171
Upgrading Cloudera Navigator HSM KMS.....	171
Setting Up an Internal Repository.....	171
Upgrading HSM KMS Using Parcels.....	171
Upgrading 5.x to 6.x Parcels for Thales HSM KMS.....	172
Upgrading HSM KMS Using Packages.....	172
Upgrading 5.x to 6.x Packages for Thales HSM KMS.....	173
Upgrading Cloudera Navigator Encrypt.....	174
Setting Up an Internal Repository.....	174
Upgrading Navigator Encrypt (RHEL-Compatible).....	174
Upgrading Navigator Encrypt (SLES).....	175
Upgrading Navigator Encrypt (Debian or Ubuntu).....	176
Best Practices for Upgrading Navigator Encrypt Hosts.....	177

Migrating from the Cloudera Manager Embedded PostgreSQL Database Server to an External PostgreSQL Database178

Prerequisites	178
Identify Roles that Use the Embedded Database Server	178
Migrate Databases from the Embedded Database Server to the External PostgreSQL Database Server	180

Configuring a Local Package Repository.....184

Creating a Permanent Internal Repository.....	184
Setting Up a Web server.....	184
Downloading and Publishing the Package Repository.....	184
Creating a Temporary Internal Repository.....	187
Configuring Hosts to Use the Internal Repository.....	188

Configuring a Local Parcel Repository.....	189
Using an Internally Hosted Remote Parcel Repository.....	189
<i>Setting Up a Web Server.....</i>	<i>189</i>
<i>Downloading and Publishing the Parcel Repository.....</i>	<i>190</i>
<i>Configuring Cloudera Manager to Use an Internal Remote Parcel Repository.....</i>	<i>192</i>
Using a Local Parcel Repository.....	193
Java Keystore and Truststore.....	194
CDH Services as TLS/SSL Servers and Clients.....	195
Certificate Formats (JKS, PEM) and Cluster Components.....	195
Recommended Keystore and Truststore Configuration.....	196
 Appendix: Apache License, Version 2.0.....	 199

Cloudera Enterprise Upgrade Guide

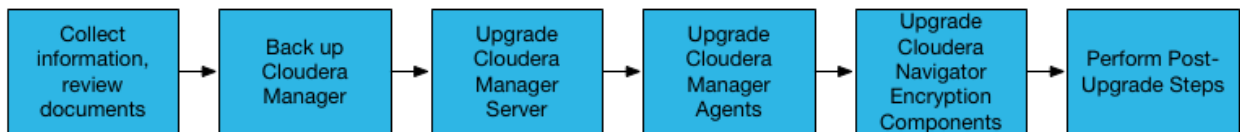
The following topics provide an overview of the Cloudera Enterprise upgrade process and include complete procedures for upgrading Cloudera Manager and your CDH clusters:

- [Upgrade Overview](#) on page 10
- [Upgrading Cloudera Manager](#) on page 38
- [Upgrading CDH](#) on page 69

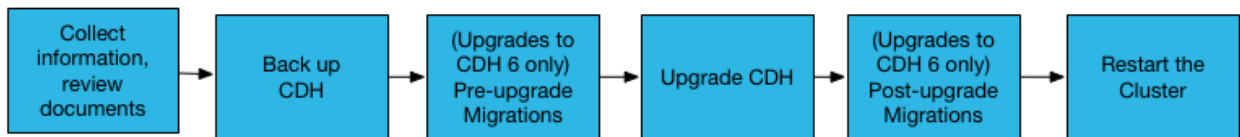
Procedures for upgrading **Cloudera Director** are discussed in the Cloudera Director documentation. See [Upgrading Cloudera Director](#).

Upgrade Overview

Upgrading consists of two major steps, upgrading Cloudera Manager and upgrading a CDH cluster.



When you upgrade Cloudera Manager, you use RPM-based package commands to upgrade the software on the Cloudera Manager server host and then Cloudera Manager manages upgrading the Cloudera Manager Agents on the remaining managed hosts. Cloudera Navigator is also upgraded when you upgrade Cloudera Manager.



When you upgrade a CDH cluster, you use Cloudera Manager to upgrade the CDH software using Cloudera *parcels* across an entire cluster or you can manually install the software on all cluster hosts using RPM-based package commands and then Cloudera Manager completes the service upgrades.

You are not required to upgrade Cloudera Manager and CDH at the same time, but the versions of Cloudera Manager and CDH must be compatible. Cloudera Manager 6.0 can manage clusters running CDH 5.7 up to CDH 5.14 and as long as the major+minor version of Cloudera Manager is equal or higher than the major+minor version of CDH. For example:

- Supported:
 - Cloudera Manager 6.0.0 and CDH 5.14.0
 - Cloudera Manager 5.14.0 and CDH 5.13.0
 - Cloudera Manager 5.13.1 and CDH 5.13.3
- Not Supported:
 - Cloudera Manager 5.14.0 and CDH 6.0.0
 - Cloudera Manager 5.12 and CDH 5.13
 - Cloudera Manager 6.0.0 and CDH 5.6

**Note:**

The [online version](#) of the **Cloudera Enterprise Upgrade Guide** allows you to create a customized version of the guide that only includes the steps required for your upgrade. Use the form at the top of pages in this guide to select your Cloudera Enterprise versions, operating system versions, databases, and other information about your upgrade. The information you enter is retained on each page in the guide.

Assessing the Impact of an Upgrade

Plan for a sufficient maintenance window to perform an upgrade. Depending on which components you are upgrading, the number of hosts in your cluster, and the type of hardware, you might need up to a full day to upgrade your cluster. Before you begin the upgrade, you need to gather some information; these steps are also detailed in the Cloudera Manager and CDH upgrade procedures.

Before upgrading, consult the release notes for Cloudera Manager and CDH to learn about API changes, deprecated features, new features, and incompatible changes.

See

- [Cloudera Enterprise 6 Release Guide](#)
- [Cloudera Enterprise 5.x Release Notes](#)

Also check the [Cloudera Enterprise 6 Requirements and Supported Versions](#) page to make sure that you are using a supported operating system, JDK, database, and other components.

Upgrades are not supported for all versions of Cloudera Enterprise. See [Supported Upgrade Paths](#) on page 20.



Important: Cloudera recommends that you test upgrades on non-production clusters before upgrading your production clusters.

There are three types of upgrades: major, minor, and maintenance:

Major Upgrade from Cloudera Manager and CDH 5.x to 6.x or higher

A major upgrade typically has the following characteristics:

- Large changes to functionality and update of Hadoop to a more recent version
- Incompatible changes in data formats
- Significant changes and additions to the user interface in Cloudera Manager
- Database schema changes for Cloudera Manager that are automatically handled by the upgrade process
- Significant down time is required to upgrade the cluster.
- [Client configurations](#) are redeployed.

Minor Upgrades

Minor upgrades upgrade your software to a higher minor version of a major release—for example from version 6.0.0 to version 6.1.0—and typically include the following:

- New functionality
- Bug fixes
- Potential database schema changes for Cloudera Manager that are handled automatically
- [Client configurations](#) are redeployed.

Incompatible changes or changes to data formats are generally not introduced in minor upgrades.

Maintenance Upgrades

Maintenance upgrades fix critical bugs or address security issues. No new functionality or incompatible changes are introduced. The version numbers for maintenance releases differ only in the third digit, for example, when upgrading from version 6.0.0 to 6.0.1.

To upgrade to a maintenance release, you only need to perform a subset of the Minor version upgrade steps. Follow the same procedures as for minor version upgrades but skip the steps that are labeled as follows:

[Not required for CDH maintenance release upgrades.]

Overview of Upgrading Cloudera Manager

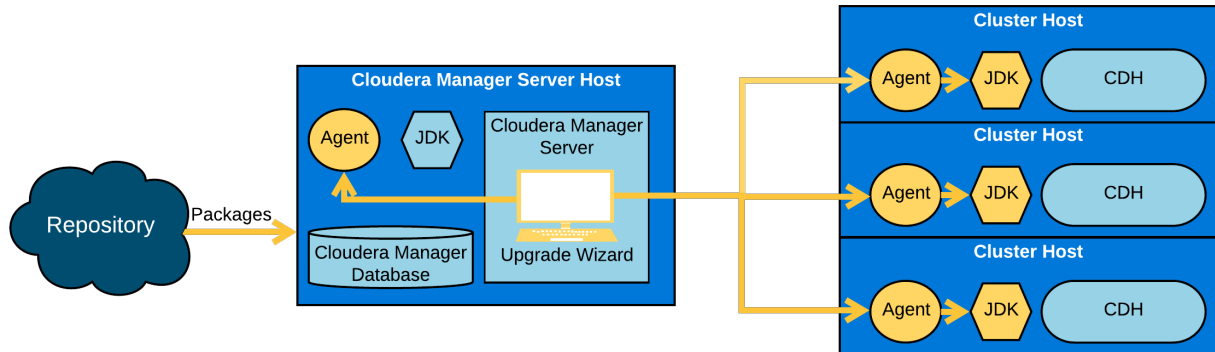


Figure 1: Cloudera Manager Upgrade

To upgrade Cloudera Manager, you perform the following tasks:

1. Back up the Cloudera Manager server databases, working directories, and several other entities. These backups can be used to restore your Cloudera Manager deployment if there are problems during the upgrade.
2. Upgrade the Cloudera Manager server software on the Cloudera Manager host using package commands from the command line (for example, `yum` on RHEL systems). Cloudera Manager automates much of this process and is recommend for upgrading and managing your CDH clusters.
3. Upgrade the Cloudera Manager agent software on all cluster hosts. The Cloudera Manager upgrade wizard can upgrade the agent software (and, optionally, the JDK), or you can install the agent and JDK software manually. The CDH software is not upgraded during this process.

For Cloudera Manager upgrade steps, see [Upgrading Cloudera Manager](#) on page 38.

If you are upgrading from Cloudera Manager 5.x to a higher version of Cloudera Manager 5.x, you can also upgrade Cloudera Manager using tarballs. See [Upgrading Cloudera Manager 5 Using Tarballs](#) for procedures to upgrade to the latest version of Cloudera Manager 5.x. (Click **Other Versions** to locate the document for earlier versions.)

Overview of Upgrading CDH

CDH upgrades contain updated versions of the Hadoop software and other components. You can use Cloudera Manager to upgrade CDH for [major, minor, and maintenance upgrades](#). The procedures vary depending on the version of Cloudera Manager you are using and the version of CDH you are upgrading to. You can use Cloudera Manager to upgrade CDH using either [parcels or packages](#).

After completing preparatory steps, you use the Cloudera Manager upgrade wizard to complete the upgrade. If you use parcels (recommended), have enabled [HDFS High Availability](#), and have a Cloudera Enterprise license, you can perform a *rolling upgrade* that does not require you to take the cluster offline during the upgrade.

For CDH upgrade steps, see [Upgrading the CDH Cluster](#) on page 92.

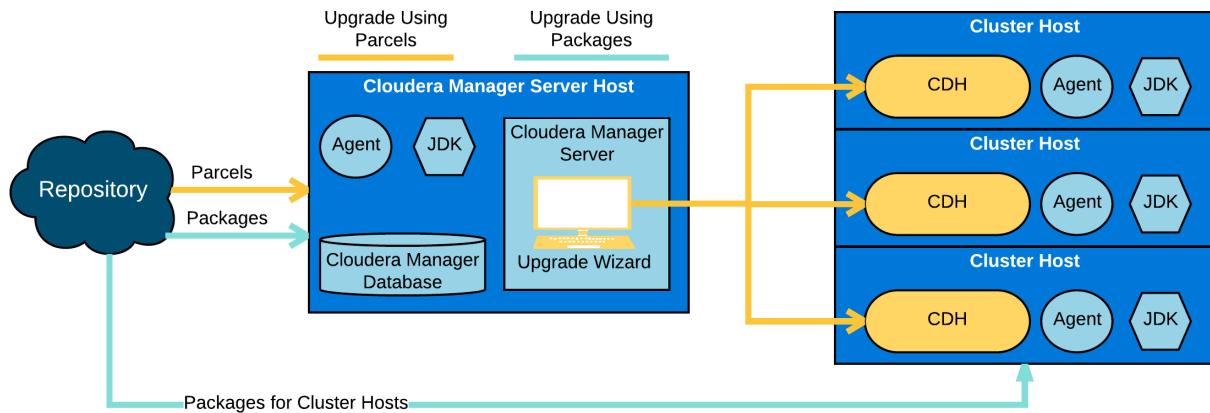


Figure 2: CDH Upgrades Using Parcels or Packages

Upgrading CDH Using Parcels (Recommended)

Upgrading CDH using parcels is the preferred method because parcels are managed by Cloudera Manager, which automatically downloads, distributes, and activates the correct versions of the software. If you have enabled high availability for HDFS, and have an Enterprise License, you can perform a rolling upgrade to upgrade CDH without cluster down time. For an easier upgrade experience, consider [switching from packages to parcels](#) so that Cloudera Manager can automate more of the process. You can also switch from packages to parcels when upgrading CDH.

Important: Rolling Upgrades are not available for upgrades to CDH 6.0.1. Cluster downtime is required for these upgrades.

Upgrading CDH Using Packages

This option is the most time consuming and requires you to log in using `ssh` and execute a series of package commands on *all hosts* in your cluster. Cloudera recommends that you instead upgrade your cluster using parcels, which allows Cloudera Manager to distribute the upgraded software to all hosts in the cluster without having to log in to each host. If you installed the cluster using packages, you can upgrade using parcels and the cluster will use parcels for subsequent upgrades.

Overview of Upgrading Cloudera Navigator Components

Cloudera Navigator Metadata and Audit servers are automatically upgraded when you upgrade Cloudera Manager. You can also optionally upgrade other Cloudera Navigator components such as Cloudera Navigator Key Trustee Server, Cloudera Navigator Key HSM, and Cloudera Navigator Encrypt. You do not have to upgrade these components along with Cloudera Manager or CDH upgrades. For compatibility information, see: [Product Compatibility Matrix for Cloudera Navigator Encryption \(Cloudera Manager 6.x\)](#) or [Product Compatibility Matrix for Cloudera Navigator Encryption \(Cloudera Manager 5.x\)](#).

See [Upgrading Cloudera Navigator Data Encryption](#) on page 152.

Install and Upgrade Notes

The notes in this topic contain important information about installing and upgrading Cloudera Enterprise. You should review these notes before installing or upgrading your software. For general release notes about Cloudera Enterprise, see [Cloudera Enterprise 6 Release Guide](#).

Upgrades to Cloudera Enterprise 6.x

Restart of Impala and Hive required for Cloudera Manager 6.2 upgrade with ADLS

After upgrading to Cloudera Manager 6.2 or higher, Impala and Hive will be marked as stale for users running CDH 6.1 and using the ADLS Service. You will need to restart Hive and Impala before being able to connect to ADLS Gen2, but all previous functionality will continue to work without a restart. The configurations that will be marked stale are:

- fs.azure.account.auth.type
- fs.azure.account.oauth.provider.type
- fs.azure.account.oauth2.client.endpoint
- fs.azure.account.oauth2.client.id
- fs.azure.account.oauth2.client.secret.

Cloudera Bug: OPSAPS-47436

TSB-359 Backup and Disaster Recovery (BDR) HDFS and Hive Replications will fail on clusters running Cloudera Manager 6.1.0

Backup and Disaster Recovery (BDR) HDFS and Hive Replications will fail when replicating from secured (Kerberized) source clusters to destination clusters that have been upgraded to Cloudera Manager 6.1.0.

This also affects new installations of Cloudera Manager 6.1.0 on the destination cluster if an admin restarts the Cloudera Manager service.

Products affected: Cloudera Manager Backup and Disaster Recovery in a secure (Kerberized) environment

Releases affected: Cloudera Manager 6.1.0 (when used as the destination cluster of HDFS and/or Hive replication)

Users affected: Customers using HDFS or Hive Replication

Severity (Low/Medium/High): High

Root Cause and Impact:

In HDFS and Hive Replication, Cloudera Manager first runs a process on the destination cluster to verify if the replication is possible. Due to a bug, the source cluster is treated as an insecure (non-kerberized) cluster. As a result, replication fails.

You will see the exception `javax.security.sasl.SaslException: GSS initiate failed [Caused by GSSException: No valid credentials provided (Mechanism level: Fail to create credential. (63) - No service creds)]` in the process stderr logs.

Immediate action required: If you use BDR, do not upgrade a destination cluster to Cloudera Manager 6.1.0. Upgrade to Cloudera Manager 6.1.1 or higher when it becomes available.

If you have already upgraded your destination cluster to Cloudera Manager to 6.1.0, use the following workaround:

1. For an existing HDFS or Hive replication schedule, select **Actions > Edit Configuration**.
2. Save the schedule.

Please note that you will need to edit only one schedule even if you have multiple schedules.

Note: This workaround is not persistent. That is, if you restart the Cloudera Manager service, you must repeat the above workaround.

Cloudera Issue: OPSAPS-48865

Fixed in Cloudera Manager 6.1.1

Upgrades from Cloudera Enterprise 5.15 or 5.16 to 6.0x are not supported

You cannot upgrade to Cloudera Manager or CDH 6.0.0 from Cloudera Manager or CDH 5.15 or 5.16.

Upgrading to CDH 6.1.0 Enables Direct SQL mode in Hive service by default

For details about the Cloudera Manager `Enable Direct SQL` option, refer to [Hive Metastore Database](#).

Upgrades from Cloudera Enterprise 6.0 Beta Release to 6.x General Release Not supported

You cannot upgrade to any Cloudera Manager or CDH 6.x general release from the Cloudera Manager or CDH 6.0 Beta release.

Cloudera Express License Enforcement

Use of Cloudera Express is limited to a total of 100 hosts running CDH6.0 or later across all environments used by an organization..

Note the following:

- Cloudera Manager will not allow you to add hosts to a CDH 6.x cluster if the total number of hosts across all CDH 6.x clusters will exceed 100.
- Cloudera Manager will not allow you to upgrade any cluster to CDH 6.x if the total number of managed CDH6.x cluster hosts will exceed 100. If an upgrade from Cloudera Manager 6.0 to 6.1 fails due to this limitation, you must downgrade Cloudera Manager to version 6.0, remove some hosts so that the number of hosts is less than 100, then retry the upgrade.



Note: If you downgrade from Cloudera Enterprise to Cloudera Express and the number of managed hosts exceeds 100, Cloudera Manager will disable all cluster management commands except for commands used to stop a cluster. You will not be able to restart or otherwise use clusters while the total number of hosts exceeds 100. Use the Cloudera Manager Admin Console to remove some hosts so that the number of hosts is less than 100.

Affected Versions: CM 6.1 and higher

Cloudera Issue: OPSAPS-46868

Cloudera Data Science Workbench is Not Supported with Cloudera Enterprise 6.0

Cloudera Data Science Workbench is not supported with Cloudera Enterprise 6.0.x. Cloudera Data Science Workbench 1.5.0 (and higher) is supported with Cloudera Manager 6.1.x (and higher) and CDH 6.1.x (and higher).

Impala roles with SELECT or INSERT privileges receive REFRESH privileges during the upgrade

Due to the Sentry and Impala fine grained privileges feature in 5.16.0, if a role has the `SELECT` or `INSERT` privilege on an object in Impala before upgrading to CDH 5.16.0, that role will automatically get the `REFRESH` privilege during the upgrade.

Hue requires manual installation of psycopg2

If you are installing or upgrading to CDH 6.0.0 and using the PostgreSQL database for the Hue database, you must install psycopg2 2.5.4 or higher on all Hue hosts. See [Installing the psycopg2 Python package](#).

Cloudera Issue: OPSAPS-47080

CDH Upgrade fails to delete Solr data from HDFS

The CDH upgrade process fails to delete Solr data from HDFS and the recreated collections fail to be initialized due to the existing indexes.

Workaround: Perform the following steps *after* you run the CDH Upgrade wizard and *before* you finalize the HDFS upgrade:

1. Log in to the Cloudera Manager Admin Console.
2. Go to the Solr service page.
3. Stop the Solr service and dependent services. Click **Actions > Stop**.
4. Click **Actions > Reinitialize Solr State for Upgrade**.
5. Click **Actions > Bootstrap Solr Configuration**.
6. Start the Solr and dependent services. Click **Actions > Start**.
7. Click **Actions > Bootstrap Solr Collections**.

Affected Versions: CDH 6.0.0

Fixed Versions: Cloudera Manager 6.0.1

Cloudera Issue: OPSAPS-47502

Package Installation of CDH Fails

When you install CDH with packages from a custom repository, ensure that the version of CDH you select for **Select the version of CDH** matches the version of CDH for the custom repository. Selecting the CDH version and specifying a custom repository are done during the **Select Repository** stage of installation.

If the versions do not match, installation fails.

Affected Versions: Cloudera Manager 6.x

Fixed Versions: N/A

Apache Issue: N/A

Cloudera Issue: OPSAPS-45703

Uninstall CDH 5 Sqoop connectors for Teradata and Netezza before upgrading to CDH 6

Sqoop includes two connectors, one for Teradata and one for Netezza. The connectors are released in separate parcels and tarballs and can be installed in Cloudera Manager or manually. The versioning of the connectors takes the form `<connector_version>c<major_cdh_version>`. For example, 1.6c5 refers to the connector 1.6 for CDH 5. The manifest files do not prohibit installing the CDH 5 connectors on CDH 6, but they are not compatible with CDH 6.

If you have CDH 5 connectors installed, they will not be automatically upgraded during the CDH upgrade, and they are not compatible with CDH 6, so they should be uninstalled before the upgrade. Keeping the CDH 5 connectors will not cause the upgrade to fail, but instead will cause a failure to occur during Sqoop runtime. Cloudera will release the connectors for CDH 6 at a later time.

For more information about the Teradata and Netezza connectors, go to [Cloudera Enterprise Connector Documentation](#) and choose the connector and version to see the documentation for your connector.

Unsupported Sqoop options cause upgrade failures

New [fail-fast](#) checks for unsupported options were introduced in CDH 6. Users should check the jobs stored in their Sqoop metastore and remove all unsupported options. Some unsupported options were silently ignored in earlier CDH versions during upgrades, but in CDH 6, the same options fail instantly. See the following JIRAs in [Apache Sqoop Incompatible Changes](#):

Upgrades to Cloudera Enterprise 5.x

Flume Kafka client incompatible changes in CDH 5.8

Due to the change of offset storage from ZooKeeper to Kafka in the CDH 5.8 Flume Kafka client, data might not be consumed by the Flume agents, or might be duplicated (if `kafka.auto.offset.reset=smallest`) during an upgrade to CDH 5.8.

Cloudera Issue: TSB-173

Upgrade to CDH 5.13 or higher Requires Pre-installation of Spark 2.1 or Spark 2.2

If your cluster has Spark 2.0 or Spark 2.1 installed and you want to upgrade to CDH 5.13 or higher, you must first upgrade to Spark 2.1 release 2 or later before upgrading CDH. To install these versions of Spark, do the following before running the CDH Upgrade Wizard:

1. Install the Custom Service Descriptor (CSD) file. See

- [Installing Spark 2.1](#)
- [Installing Spark 2.2](#)



Note:

Spark 2.2 requires that JDK 1.8 be deployed throughout the cluster. JDK 1.7 is not supported for Spark 2.2.

See [Step 2: Install Java Development Kit](#).

2. Download, distribute, and activate the Parcel for the version of Spark that you are installing:

- **Spark 2.1 release 2:** The parcel name includes "cloudera2" in its name.
- **Spark 2.2 release 1:** The parcel name includes "cloudera1" in its name.

See [Managing Parcels](#).

Affected versions: CDH 5.13.0 and higher

Cloudera Issue: CDH-56775

Sentry may require increased Java heap settings before upgrading CDH to 5.13

Before upgrading to CDH 5.13 or higher, you may need to increase the size of the Java heap for Sentry. A warning will be displayed during upgrade, but it is the user's responsibility to ensure this setting is adjusted properly before proceeding. See [Performance Guidelines](#).

Affected versions: CDH 5.13 or higher

Cloudera Issue: OPSAPS-42541

Apache MapReduce Jobs May Fail During Rolling Upgrade to CDH 5.11.0 or CDH 5.11.1

In CDH 5.11, Cloudera introduced four new counters that are reported by MapReduce jobs. During a rolling upgrade from a cluster running CDH 5.10.x or lower to CDH 5.11.0 or CDH 5.11.1, a MapReduce job with an application master running on a host running CDH 5.10.x or lower may launch a map or reduce task on one of the newly-upgraded CDH 5.11.0 or CDH 5.11.1 hosts. The new task will attempt to report the new counter values, which the old application master will not understand, causing an error in the logs similar to the following:

```
2017-06-08 17:43:37,173 WARN [Socket Reader #1 for port 41187]  
org.apache.hadoop.ipc.Server: Unable to read call parameters for client 10.17.242.22 on  
connection protocol org.apache.hadoop.mapred.TaskUmbilicalProtocol for rpcKind
```

```
RPC_WRITABLE
java.lang.ArrayIndexOutOfBoundsException: 23
    at
    ...
```

This error could cause the task and the job to fail.

Workaround:

Avoid performing a rolling upgrade to CDH 5.11.0 or CDH 5.11.1 from CDH 5.10.x or lower. Instead, skip CDH 5.11.0 and CDH 5.11.1 if you are performing a rolling upgrade, and upgrade to CDH 5.12 or higher, or CDH 5.11.2 or higher when the release becomes available.

Cloudera Issue: DOCS-2384, TSB-241

Cloudera Manager set catalogd default jvm memory to 4G can cause out of memory error on upgrade to Cloudera Manager 5.7 or higher

After upgrading to 5.7 or higher, you might see a reduced Java heap maximum on Impala Catalog Server due to a change in its default value. Upgrading from Cloudera Manager lower than 5.7 to Cloudera Manager 5.8.2 no longer causes any effective change in the Impala Catalog Server Java Heap size.

When upgrading from Cloudera Manager 5.7 or later to Cloudera Manager 5.8.2, if the Impala Catalog Server Java Heap Size is set at the default (4GB), it is automatically changed to either 1/4 of the physical RAM on that host, or 32GB, whichever is lower. This can result in a higher or a lower heap, which could cause additional resource contention or out of memory errors, respectively.

Cloudera Issue: OPSAPS-34039

Supported Upgrade Paths

This page describes the supported upgrade paths for Cloudera Enterprise.

Supported CDH Upgrade Paths

Current Cloudera Manager Version	Supported Cloudera Manager Upgrade Paths	Supported CDH Upgrade Paths
6.2	<ul style="list-style-type: none"> 6.2.x maintenance releases 	<ul style="list-style-type: none"> 6.2.x maintenance releases 6.1.x 6.0.x Any minor version of CDH 5.7 - 5.16
6.1	<ul style="list-style-type: none"> 6.2.x 6.1.x maintenance releases 	<ul style="list-style-type: none"> 6.1.x maintenance releases 6.0.x Any minor version of CDH 5.7 - 5.16
6.0	<ul style="list-style-type: none"> 6.1.x 6.0.x maintenance releases 	<ul style="list-style-type: none"> 6.0.x maintenance releases Any minor version of CDH 5.7 - 5.14
6.0 Beta release	No supported upgrades.	No supported upgrades.
5.16	<ul style="list-style-type: none"> 6.2.x 6.1.x 5.16.x maintenance releases 	<ul style="list-style-type: none"> 5.16.x maintenance releases Any minor version of that is lower or equal to the Cloudera Manager version.
5.15	<ul style="list-style-type: none"> 6.2.x 6.1.x 5.16.x 5.15.x maintenance releases 	<ul style="list-style-type: none"> 5.15.x maintenance releases Any minor version of CDH 5 that is lower or equal to the Cloudera Manager version.
5.7 - 5.14	<ul style="list-style-type: none"> 6.2.x 6.1.x 6.0 5.7.x - 5.16 	<ul style="list-style-type: none"> Any minor version of CDH that is lower or equal to the Cloudera Manager version.
5.0 - 5.6	<ul style="list-style-type: none"> 5.16 or lower 	<ul style="list-style-type: none"> Any minor version of CDH 5.0 - 5.6 that is lower or equal to the Cloudera Manager version.

Cloudera Manager support for CDH

The versions of CDH clusters that can be managed by Cloudera Manager are limited to the following:

Cloudera Manager Version	Supported CDH versions
6.2	<ul style="list-style-type: none"> CDH 6.2

Cloudera Manager Version	Supported CDH versions
	<ul style="list-style-type: none">• CDH 6.1• CDH 6.0• CDH 5.7 - 5.16
6.1	<ul style="list-style-type: none">• CDH 6.1• CDH 6.0• CDH 5.7 - 5.16
6.0	<ul style="list-style-type: none">• CDH 6.0• CDH 5.7 - 5.14
5.16.x or lower	Any minor version of CDH 5.0 - 5.16 that is lower or equal to the Cloudera Manager version.

Upgrading the JDK

Cloudera Manager and CDH require a supported version of the Java Development Kit (JDK) to be installed on all hosts. For details, see [Java Requirements](#)

Table 1: Supported JDKs

Cloudera Enterprise Version	Supported JDK
5.3 -5.15	Oracle JDK 1.7, Oracle JDK 1.8
5.16 and higher 5.x releases	Oracle JDK 1.7, Oracle JDK 1.8, OpenJDK 1.8
6.0	Oracle JDK 1.8
6.1	Oracle JDK 1.8, OpenJDK 1.8
6.2	Oracle JDK 1.8, OpenJDK 1.8



Warning:

- If you are upgrading from a lower major version of the JDK to JDK 1.8 or from JDK 1.6 to JDK 1.7, and you are using AES-256 bit encryption, you must install new encryption policy files. (In a Cloudera Manager deployment, you automatically install the policy files; for unmanaged deployments, install them manually.) See [Using AES-256 Encryption](#) on page 27. This step is not required when using JDK 1.8.0_162 or greater. JDK 1.8.0_162 enables unlimited strength encryption by default.

You must also ensure that the Java Truststores are retained during the upgrade. (See [Recommended Keystore and Truststore Configuration](#) on page 196.)

There are two procedures you can use to upgrade the Oracle JDK:

- [Installing JDK 8 as part of an Upgrade to Cloudera Manager 6.x](#)

If you are upgrading to Cloudera Manager 6.0.0 or higher, you can manually install JDK 1.8 on the Cloudera Manager server host, and then, as part of the Cloudera Manager upgrade process, you can specify that Cloudera Manager upgrade the JDK on the remaining hosts.



Note: Cloudera Manager only installs Oracle JDK. You can upgrade to OpenJDK using these steps.

- [Manually Installing Oracle JDK 1.8 on page 23](#)

You can manually install JDK 1.8 on all managed hosts. If you are upgrading to any version of Cloudera Manager 5.x, you must use this procedure. Continue with the steps in the next section.

To upgrade to OpenJDK, see [Manually Migrating from Oracle JDK to OpenJDK](#) on page 25

Manually Installing Oracle JDK 1.8



Important: Manual upgrade of Oracle JDK 1.8 requires down time to stop and restart your cluster.

You can manually install Oracle JDK 1.8 on all managed hosts. If you are upgrading to any version of Cloudera Manager 5.x, you must use the following procedure:

1. Download the `.tar.gz` file for one of the 64-bit versions of Oracle JDK 1.8 from [Java SE 8 Downloads](#). (This link is correct at the time of writing, but can change.) See [Supported Java versions for CDH 5](#) or [Java Requirements](#).
2. Perform the following steps on all hosts that you are upgrading:
 - a. Log in to the host as `root` using `ssh`.
 - b. Copy the downloaded `.tar.gz` file to the host.
 - c. Extract the JDK to the folder `/usr/java/jdk-version`. For example:

```
tar xvfz /path/to/jdk-8u<update_version>-linux-x64.tar.gz -C /usr/java/
```

3. If you have [configured TLS for Cloudera Manager](#), copy the `jssecacerts` file from the previous JDK installation to the new JDK installation. This step is not required when using JDK 1.8.0_162 or greater. JDK 1.8.0_162 enables unlimited strength encryption by default.

For example:

```
cp previous_java_home/jre/lib/security/jssecacerts new_java_home/jre/lib/security
```

(Substitute `previous_java_home` and `new_java_home` with the paths to the JDK installations.)

4. Configure the location of the JDK on cluster hosts.
 - a. Open the Cloudera Manager Admin Console.
 - b. In the main navigation bar, click the **Hosts** tab. If you are configuring the JDK location on a specific host only, click the link for that host.
 - c. Click the **Configuration** tab.
 - d. Select **Category > Advanced**.
 - e. Set the **Java Home Directory** property to the custom location.
 - f. Click **Save Changes**.
5. On the Cloudera Manager Server host only (not required for other hosts):
 - a. Open the file `/etc/default/cloudera-scm-server` in a text editor.
 - b. Edit the line that begins with `export JAVA_HOME` (if this line does not exist, add it) and change the path to the path of the new JDK (you can find the path under `/usr/java`).

For example: (RHEL and SLES)

```
export JAVA_HOME="/usr/java/jdk1.8.0_141-cloudera"
```

For example: (Ubuntu)

```
export JAVA_HOME="/usr/lib/jvm/java-8-oracle-cloudera"
```

- c. Save the file.
- d. Restart the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl restart cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

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

6. Restart the Cloudera Management Service.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters** > **Cloudera Management Service**.
- c. Select **Actions** > **Restart**.

7. Restart all clusters:

- a. On the **Home** > **Status** tab, click



to the right of the cluster name and select **Restart**.

- b. Click **Restart** that appears in the next screen to confirm. If you have enabled [high availability for HDFS](#), you can choose [Rolling Restart](#) instead to minimize cluster downtime. The **Command Details** window shows the progress of stopping services.

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

8. Delete the files from your previous Java installation. If you do not delete these files, Cloudera Manager and other components may continue to use the old version of the JDK.

OpenJDK

You must [install a supported version](#) of OpenJDK. If your deployment uses a version of OpenJDK lower than 1.8.0_181, see [TLS Protocol Error with OpenJDK](#).

Manually Installing OpenJDK

Before installing Cloudera Manager and CDH, perform the steps in this section to install [OpenJDK](#) on all hosts in your cluster(s).

When you install Cloudera Enterprise, Cloudera Manager includes an option to install Oracle JDK. De-select this option.

See [Supported JDKs](#) for information on which JDK versions are supported for Cloudera Enterprise releases.

You must [install a supported version](#) of OpenJDK. If your deployment uses a version of OpenJDK lower than 1.8.0_181, see [TLS Protocol Error with OpenJDK](#).



Note: If you intend to enable [Auto-TLS](#), note the following:

You can specify a PEM file containing trusted CA certificates to be imported into the Auto-TLS truststore. If you want to use the certificates in the cacerts truststore that comes with OpenJDK, you must convert the truststore to PEM format first. However, OpenJDK ships with some intermediate certificates that cannot be imported into the Auto-TLS truststore. You must remove these certificates from the PEM file before importing the PEM file into the Auto-TLS truststore. This is not required when upgrading to OpenJDK from a cluster where Auto-TLS has already been enabled.

Log in to each host and run the following command:

RHEL

```
su -c yum install java-1.8.0-openjdk-devel
```

Ubuntu

```
sudo apt-get install openjdk-8-jdk
```

SLES

```
sudo zypper install java-1_8_0-openjdk-devel
```

Manually Migrating from Oracle JDK to OpenJDK

If you have Oracle JDK installed on the hosts managed by Cloudera Manager, use the steps in this section to migrate your deployment to use [OpenJDK](#). The steps below require you to restart all clusters, which will cause downtime as the hosts restart. If your clusters have enabled [high availability for HDFS](#), you can use a rolling restart to restart the clusters without downtime. Note that until the rolling restart completes, some of the hosts in your cluster will still be using the Oracle JDK. If you do not want a temporarily mixed environment, you can stop the cluster before performing the steps in this section to migrate the JDK.

You must [install a supported version](#) of OpenJDK. If your deployment uses a version of OpenJDK lower than 1.8.0_181, see [TLS Protocol Error with OpenJDK](#).

1. Find out the package name of your currently installed Oracle JDK by running the following commands. The grep commands attempt to locate the installed JDK. If the JDK package is not returned, try looking for the string `sdk`.

RHEL

```
yum list installed |grep oracle
```

Ubuntu

```
apt list --installed | grep oracle
```

SLES

```
zypper search --installed-only |grep oracle
```

The command will return values similar to the following example::

```
oracle-j2sdk1.7.x86_64          1.7.0+update67-1          @cloudera-manager
```

The Oracle JDK package name in the above example is: `oracle-j2sdk1.7.x86_64`.

2. Locate the version of Java that is installed: Open the Cloudera Manager Admin Console and go to **Support > About**. The Java version displays along with other version information.
3. Log in to each host managed by Cloudera Manager (including the Cloudera Manager server host) and run the following command to install OpenJDK:

RHEL

```
su -c yum install java-1.8.0-openjdk-devel
```

Ubuntu

```
sudo apt-get install openjdk-8-jdk
```

SLES

```
sudo zypper install java-1_8_0-openjdk-devel
```

4. On the Cloudera Manager Server host only (not required for other hosts):

- a. Open the file `/etc/default/cloudera-scm-server` in a text editor.
- b. Edit the line that begins with `export JAVA_HOME` (if this line does not exist, add it) and change the path to the path of the new JDK (the JDK is usually installed in `/usr/lib/jvm/`)(or `/usr/lib64/jvm` on SLES 12), but the path may differ depending on how the JDK was installed).

For example:

RHEL 6

```
export JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk.x86_64"
```

RHEL 7

```
export JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk"
```

Debian

```
export JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64"
```

Ubuntu

```
export JAVA_HOME="/usr/lib/jvm/openjdk-8-jdk"
```

SLES

```
export JAVA_HOME="/usr/lib64/jvm/java-1.8.0-openjdk"
```

- c. Save the file.
- d. Restart the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl restart cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

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

5. Restart the **Cloudera Management Service**.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters > Cloudera Management Service**.
- c. Select **Actions > Restart**.

6. Restart all clusters:

- a. On the **Home > Status** tab, click



to the right of the cluster name and select either **Restart** or **Rolling Restart**. Selecting [Rolling Restart](#) minimizes cluster downtime and is available only if you have enabled [high availability for HDFS](#).

- b. Click **Restart** or **Rolling Restart** that appears in the next screen to confirm. The **Command Details** window shows the progress of stopping services.

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

7. Remove the Oracle JDK:

- a. Run the following command using the Java version to verify the current running version of Java:

```
/usr/java/jdk<Java version string>-cloudera/bin/java -version
```

- b. Perform the following steps on all hosts managed by Cloudera Manager:

- a. Run the following command to remove the Oracle JDK: (If you do not delete these files, Cloudera Manager and other components may continue to use the old version of the JDK.)

RHEL

```
yum remove oracle-<JDK package name>
```

Ubuntu

```
apt-get remove oracle-<JDK package name>
```

SLES

```
zypper rm oracle-<JDK package name>
```

- b. Confirm that the package has been removed:

RHEL

```
yum list installed |grep -i oracle
```

Ubuntu

```
apt list --installed | grep -i oracle
```

SLES

```
zypper search --installed-only |grep -i oracle
```

Using AES-256 Encryption



Note: This step is not required when using JDK 1.8.0_162 or greater. JDK 1.8.0_162 enables unlimited strength encryption by default.

If you are using CentOS/Red Hat Enterprise Linux 5.6 or higher, or Ubuntu, which use AES-256 encryption by default for tickets, you must install the **Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy File** on all cluster and Hadoop user machines. For JCE Policy File installation instructions, see the `README.txt` file included in the `jce_policy-x.zip` file. For more information, see [Java Cryptography Extension \(JCE\) Unlimited Strength Jurisdiction Policy File](#)

Alternately, you can configure Kerberos to not use AES-256 by removing `aes256-cts:normal` from the `supported_etypes` field of the `kdc.conf` or `krb5.conf` file. After changing the `kdc.conf` file, you must restart both the KDC and the kadmin server for those changes to take affect. You may also need to re-create or change the password of the relevant principals, including, potentially the Ticket Granting Ticket principal (`krbtgt/REALM@REALM`). If AES-256 is still used after completing steps, the `aes256-cts:normal` setting existed when the Kerberos database was created. To fix this, create a new Kerberos database and then restart both the KDC and the kadmin server.

To verify the type of encryption used in your cluster:

1. On the local KDC host, type this command to create a test principal:

```
kadmin -q "addprinc test"
```

2. On a cluster host, type this command to start a Kerberos session as the test principal:

```
kinit test
```

3. On a cluster host, type this command to view the encryption type in use:

```
klist -e
```

If AES is being used, output like the following is displayed after you type the `klist` command; note that AES-256 is included in the output:

```
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@SCM
Valid starting    Expires            Service principal
05/19/11 13:25:04 05/20/11 13:25:04  krbtgt/SCM@SCM
      Etype (skey, tkt): AES-256 CTS mode with 96-bit SHA-1 HMAC, AES-256 CTS mode with
      96-bit SHA-1 HMAC
```

Configuring a Custom Java Home Location



Note: Cloudera strongly recommends installing Oracle JDK at `/usr/java/<jdk-version>` and OpenJDK at `/usr/lib/jvm` (or `/usr/lib64/jvm` on SLES 12), which allows Cloudera Manager to auto-detect and use the correct JDK version. If you install the JDK anywhere else, you must follow these instructions to configure Cloudera Manager with your chosen location. The following procedure changes the JDK location for Cloudera Management Services and CDH cluster processes only. It does not affect the JDK used by other non-Cloudera processes.

Although not recommended, the Java Development Kit (JDK), which Cloudera services require, may be installed at a custom location if necessary. These steps assume you have already installed the JDK as documented in [Step 2: Install Java Development Kit](#) or as part of an upgrade.

To modify the Cloudera Manager configuration to ensure the JDK can be found:

1. Open the Cloudera Manager Admin Console.
2. In the main navigation bar, click the **Hosts** tab. If you are configuring the JDK location on a specific host only, click the link for that host.
3. Click the **Configuration** tab.
4. Select **Category > Advanced**.
5. Set the **Java Home Directory** property to the custom location.
6. Click **Save Changes**.
7. Restart all services.

Upgrading the Operating System

This topic describes the additional steps needed to upgrade the operating system of a host managed by Cloudera Manager to a higher version, including major and minor releases.

Upgrading the operating system to a higher version but within the same major release is called a **minor release** upgrade. For example, upgrading from Redhat 6.8 to 6.9. This is a relatively simple procedure that involves properly shutting down all the components, performing the operating system upgrade, and then restarting everything in reverse order.

Upgrading the operating system to a different major release is called a **major release upgrade**. For example, upgrading from Redhat 6.8 to 7.4. This is a much more complex procedure to do it in-place, and some operating systems do not support these upgrades. Therefore, the procedures for upgrading specific operating systems are not covered in this topic.

This topic primarily describes all the additional steps such as backing up essential files and removing and reinstalling all the necessary Cloudera Enterprise packages and parcels.

Getting Started with Operating System Upgrades

Prerequisites

- Ensure that the versions of Cloudera Manager and CDH supports your new operating system.
 - See [Operating System Requirements](#) when using Cloudera Manager 5.x.
 - See [Operating System Requirements](#) when using Cloudera Manager 6.x.
- If you are using unsupported versions, see [Upgrade Cloudera Manager](#) or [Upgrade CDH](#).
- Ensure that the host has access to the Cloudera Manager server, daemon and agent packages that are supported for the new operating system, either by having access to <https://archive.cloudera.com> or a [local package repository](#).
- Ensure that the Cloudera Manager server has access to the parcels that are using supported for the new Operating System, either by having access to <https://archive.cloudera.com> or a [local parcel repository](#).
- If you have a patched package or parcel installed, make sure you have the same package or parcel for the new Operating System and it has been made available to Cloudera Manager.
- Understand that performing a major release upgrade for the operating system in-place may be quite tricky and risky.

Backing Up Host Files Before Upgrading the Operating System

This topic describes how to backup important files on your host before upgrading the operating system.

Backing Up

1. Create a top-level backup directory.

```
export CM_BACKUP_DIR=`date +%F`-CM"
echo $CM_BACKUP_DIR
mkdir -p $CM_BACKUP_DIR
```

2. Back up the Agent directory and the runtime state.

```
sudo -E tar -cf $CM_BACKUP_DIR/cloudera-scm-agent.tar --exclude=*.sock
/etc/cloudera-scm-agent /etc/default/cloudera-scm-agent /var/run/cloudera-scm-agent
/var/lib/cloudera-scm-agent
```

3. Back up the Cloudera Manager Server directories:

```
sudo -E tar -cf $CM_BACKUP_DIR/cloudera-scm-server.tar /etc/cloudera-scm-server  
/etc/default/cloudera-scm-server
```



Note: Backup is recommended but not always required for a minor release upgrade.

Before You Upgrade the Operating System

This topic describes steps you must perform before upgrading the operating system on a host managed by Cloudera Manager.

Decommission and Stop Running Roles

1. Log in to the **Cloudera Manager Admin Console**.
2. From the **All Hosts** page, select the host that you wish to upgrade. Cloudera recommends that you upgrade only one host at a time.
3. Select **Begin Maintenance (Suppress Alerts/Decommission)** from the **Actions** menu.
4. Select **Host Decommission** from the **Actions** menu. Any roles that do not require decommission will be skipped.
5. If the operating system upgrade procedure takes less than 30 minutes per node, you do not need to decommission the DataNode.

If the Cloudera Manager and CDH version are both 5.14 or greater, you can also choose the **Take DataNode Offline** feature.

If in doubt, decommission the roles.

- When a DataNode is decommissioned, the NameNode ensures that every block from the DataNode is still available across the cluster as specified by the replication factor. This procedure involves copying blocks off the DataNode in small batches. In cases where a DataNode has several thousand blocks, decommissioning takes several hours.
- When a DataNode is turned off without being decommissioned:
 - The NameNode marks the DataNode as dead after a default of 10m 30s (controlled by the `dfs.heartbeat.interval` and `dfs.heartbeat.recheck.interval` configuration properties).
 - The NameNode schedules the missing replicas to be placed on other DataNodes.
 - When the DataNode comes back online and reports to the NameNode, the NameNode schedules blocks to be copied to it while other nodes are decommissioned or when new files are written to HDFS.
- You can also speed up the decommissioning of a DataNode by increasing values for these properties:
 - `dfs.max-repl-streams`: The number of simultaneous streams used to copy data.
 - `dfs.balance.bandwidthPerSec`: The maximum amount of bandwidth that each DataNode can utilize for balancing, in bytes per second.
 - `dfs.namenode.replication.work.multiplier.per.iteration`: NameNode configuration requiring a restart, defaults to 2 but can be raised to 10 or higher.

This determines the total amount of block transfers to begin in parallel at a DataNode for replication, when such a command list is being sent over a DataNode heartbeat by the NameNode. The actual number is obtained by multiplying this value by the total number of live nodes in the cluster. The result number is the number of blocks to transfer immediately, per DataNode heartbeat.

6. Once that is completed, select the same host again and choose **Stop Roles on Hosts**.



Warning: If you have not enabled high availability for HDFS, HBase, MapReduce, YARN, Oozie, or Sentry, stopping the running single master role will cause an outage for that service. Specifically, slave roles on other hosts will stop abruptly. Cloudera recommends that you stop these services prior to the host upgrade.



Important: When upgrading hosts that are part of a ZooKeeper quorum, ensure that the majority of the quorum is still available.

Stop Cloudera Manager Agent

1. Hard Stop the Cloudera Manager Agent.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop supervisord
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent hard_stop
```



Important: This will ask you to confirm with `hard_stop_confirmed` because this will terminate any Hadoop services on the host (if any) unconditionally.

Stop Cloudera Manager Server & Agent

1. Hard Stop the Cloudera Manager Agent.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop supervisord
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent hard_stop
```



Important: This will ask you to confirm with `hard_stop_confirmed` because this will terminate any Hadoop services on the host (if any) unconditionally.

2. Stop the Cloudera Management Service.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters** > **Cloudera Management Service**.
- c. Select **Actions** > **Stop**.

3. Stop the Cloudera Manager Server.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server stop
```

Stop Databases

1. If you are using the embedded PostgreSQL database, stop the Cloudera Manager Embedded PostgreSQL database:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server-db
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server-db stop
```

If you are not using the embedded PostgreSQL database and you attempt to stop it, you might see a message indicating that the service cannot be found. If you see a message that the shutdown failed, then the embedded database is still running, probably because services are connected to the Hive metastore. If the database shutdown fails due to connected services, issue the following command:

RHEL-compatible 7 and higher, Ubuntu 16.04

```
sudo service cloudera-scm-server-db next_stop_fast  
sudo service cloudera-scm-server-db stop
```

All other Linux distributions

```
sudo service cloudera-scm-server-db fast_stop
```

2. If there are other database servers running on this host, they must be stopped also.

Remove Packages & Parcels

Packages for the older operating system won't be able to start on the new operating system. Remove old packages from the host.

1. RHEL / CentOS

```
sudo yum remove cloudera-manager-daemons cloudera-manager-agent  
cloudera-manager-server-db-2
```

SLES

```
sudo zypper remove cloudera-manager-daemons cloudera-manager-agent  
cloudera-manager-server-db-2
```

Debian / Ubuntu

```
sudo apt-get purge cloudera-manager-daemons cloudera-manager-agent  
cloudera-manager-server-db-2
```

2. RHEL / CentOS

```
sudo yum remove cloudera-manager-server cloudera-manager-daemons cloudera-manager-agent  
cloudera-manager-server-db-2
```

SLES

```
sudo zypper remove cloudera-manager-server cloudera-manager-daemons cloudera-manager-agent  
cloudera-manager-server-db-2
```


Debian / Ubuntu

```
sudo apt-get purge cloudera-manager-server cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

3. Remove old CDH parcels from the host. These are built for your old operating system.

The Cloudera Manager agent will download and activate the proper parcel for the new operating system when it is started.

Empty the contents of the following directories. These are the defaults for parcel storage - if you use other directories, please change accordingly.

```
sudo rm -rf /opt/cloudera/parcels/*
```

```
sudo rm -rf /opt/cloudera/parcel-cache/*
```

Upgrade the Operating System

Important: When there are no Hadoop services or Cloudera Manager roles running from this host, you may proceed to upgrade the operating system of this host, make sure to leave the data partitions (for example, dfs.data.dir) unchanged.

Use the operating system upgrade procedures provided by your operating system vendor (for example: RedHat or Ubuntu) to download their software and perform the operating system upgrade.

After You Upgrade the Operating System

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

This topic describes how to upgrade the operating system on a Cloudera Manager managed host.

Establish Access to the Software

Cloudera Manager needs access to a package repository that contains the updated software packages.

- If the Cloudera Manager hosts have internet access, you can use the publicly available repositories from <https://archive.cloudera.com>. Modify the sample repo files below for your operating system and version.
- If the Cloudera Manager hosts *do not* have internet access, configure [a local package repository](#) hosted on your network. For example: `http://MyWebServer:1234/cloudera-repos`
- Replace `archive.cloudera.com` in the sample repo files below with the URL for your local repository.

Package Repository URL:

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Remove any older files in the existing repository directory:

RHEL / CentOS

```
sudo rm /etc/yum.repos.d/cloudera*manager.repo*
```

SLES

```
sudo rm /etc/zypp/repos.d/cloudera*manager.repo*
```

Debian / Ubuntu

```
sudo rm /etc/apt/sources.list.d/cloudera*.list*
```

3. Fill in the form at the top of this page.

4. Create a repository file so that the package manager can locate and download the binaries:

RHEL / CentOS

Create a file named `/etc/yum/repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
baseurl=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

SLES

Create a file named `/etc/zypp/repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
baseurl=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

Debian / Ubuntu

Create a file named `/etc/apt/sources.list.d/cloudera_manager.list` with the following content:

```
# Packages for Cloudera Manager
deb https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
deb-src https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
```

```
sudo apt-get update
```

The repository file, as created, refers to the *most recent* maintenance release of the specified minor release. If you would like to use a *specific* maintenance version, for example 5.15.1, replace 5.15 with 5.15.1 in the generated repository file shown above.

5. A Cloudera Manager upgrade can introduce new package dependencies. Your organization may have restrictions or require prior approval for installation of new packages. You can determine which packages may be installed or upgraded:

RHEL / CentOS

```
yum deplist cloudera-manager-agent
```

SLES

```
zypper info --requires cloudera-manager-agent
```

Debian / Ubuntu

```
apt-cache depend cloudera-manager-agent
```

Reinstall Cloudera Manager Daemon & Agent Packages

Re-install the removed Cloudera packages.

1. Install the agent packages. Include the `cloudera-manager-server-db-2` package in the command only if you are using the embedded PostgreSQL database.

RHEL / CentOS

```
sudo yum clean all
sudo yum install cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

SLES

```
sudo zypper clean --all
sudo zypper install cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

Debian / Ubuntu

```
sudo apt-get clean
```

```
sudo apt-get update
```

```
sudo apt-get install cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

Verify that the configuration files (that were backed up) are intact. Correct if necessary.

Reinstall Cloudera Manager Server, Daemon & Agent Packages

Re-install the removed Cloudera packages.

1. Install the packages. Include the `cloudera-manager-server-db-2` package in the command only if you are using the embedded PostgreSQL database.

RHEL / CentOS

```
sudo yum clean all
sudo yum install cloudera-manager-server cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

SLES

```
sudo zypper clean --all
sudo zypper install cloudera-manager-server cloudera-manager-daemons
cloudera-manager-agent cloudera-manager-server-db-2
```

Debian / Ubuntu

```
sudo apt-get clean
```

```
sudo apt-get update
```

```
sudo apt-get install cloudera-manager-server cloudera-manager-daemons  
cloudera-manager-agent cloudera-manager-server-db-2
```

Verify that the configuration files (that were backed up) are intact. correct if necessary.

Edit Cloudera repository file to point to the repositories designed for your new operating system.

Start Databases

1. If you are using the embedded PostgreSQL database, start the database:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server-db
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server-db start
```

2. If there were database servers stopped, they must be restarted.

Start Cloudera Manager Server & Agent

The appropriate services typically will start automatically on reboot. Otherwise, start the Cloudera Manager Server & Agent as necessary.

1. Start the rpcbind service if it is not automatically started.

```
sudo service rpcbind start
```

2. Start the **Cloudera Manager Agent**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-agent
```

If the agent starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent start
```

You should see the following:

```
Starting cloudera-scm-agent: [ OK ]
```

3. Start the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server
```

If the Cloudera Manager Server starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server start
```

You should see the following:

```
Starting cloudera-scm-server: [ OK ]
```

4. Verify that the Cloudera Manager Agent downloaded a proper parcel for your new operating system. You can use the following command to check in Cloudera Manager logs for downloaded parcels:

```
grep "Completed download" /var/log/cloudera-scm-agent/cloudera-scm-agent.log
```

(Download might take some time. Look for the operating system in the names of the downloaded parcels.)

Start Roles

1. From the **All Hosts** page, select the host that you have just upgraded.
2. Choose **End Maintenance (Enable Alerts/Decommission)** from the **Actions** menu and confirm.
3. Start any Cloudera Management Service roles that were running on this host and were stopped.
4. Choose **Host Recommission** from the **Actions** menu and confirm.
5. Choose **Start Roles on Hosts** from the **Actions** menu and confirm.
6. Start any services that were stopped due to lack of high availability.

Upgrading Cloudera Manager

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

This topic describes how to upgrade Cloudera Manager from any 5.x or 6.x version to a higher version of Cloudera Manager 5.x or 6.x, including major, minor, and maintenance releases using operating system command-line package commands, and then complete the upgrade using Cloudera Manager. Upgrades to Cloudera Manager 6.x are only possible from Cloudera Manager 5.7 or higher.

There are other limitations on the versions of Cloudera Manager and CDH that are eligible for upgrades. See [Supported Upgrade Paths](#) on page 20.

Cloudera Navigator is also upgraded when you upgrade Cloudera Manager.

The Cloudera Manager upgrade process does the following:

- Upgrades the database schema to reflect the current version.
- Upgrades the Cloudera Manager Server and all supporting services.
- Upgrades the Cloudera Manager agents on all hosts.
- Redeploys client configurations to ensure that client services have the most current configuration.
- Upgrades Cloudera Navigator.

Upgrading Cloudera Manager does not upgrade CDH clusters. See [Upgrading CDH](#) on page 69 for CDH upgrade procedures.

Getting Started Upgrading Cloudera Manager

Before you upgrade Cloudera Manager, you need to gather some information and review the limitations and release notes. Fill in the **My Environment** form below to customize your Cloudera Manager upgrade procedures. See the [Collect Information](#) section below for assistance in locating the required information.



Warning: Redhat 5 is not a supported operating system for Cloudera Manager 6.x.



Warning: SLES 11 is not a supported operating system for Cloudera Manager 6.x.



Warning: Ubuntu 14 and lower are not supported operating systems for Cloudera Manager 6.x.



Warning: Debian is not a supported operating system for Cloudera Manager 6.x.



Warning: Ubuntu 12 is not a supported operating systems for Cloudera Manager 5.16 and higher.



Important: Oracle 11 is not supported in Cloudera Manager 6.x. If your deployment uses Oracle 11 as the database for Cloudera Manager, you must upgrade to a [supported database](#) before upgrading Cloudera Manager.

**Warning:**

If you have Cloudera Manager Backup and Disaster Recovery replication schedules configured, note the following:

Due to a bug, after you upgrade to Cloudera Manager 6.1.0 on a destination cluster, existing Backup and Disaster Recovery replication schedules will fail if the source cluster uses Kerberos authentication.

To resolve this issue, use the following workaround:

1. Select an existing HDFS or Hive replication schedule, and click **Actions > Edit Configuration**.
2. Save the schedule.

If you restart the Cloudera Manager service, you need to repeat the workaround. A fix for this bug is available in the Cloudera Manager 6.1.1 maintenance release.



Important: Cloudera Manager 6.0 supports upgrading from Cloudera Manager 5.7 or higher, to Cloudera Manager 5.7 through 5.14. All clusters managed by Cloudera Manager must be running CDH 5.7 or higher.



Warning: The selected version of Cloudera Manager is too old and cannot be upgraded to 6.x. Upgrade your deployment to Cloudera Manager 5.7 or higher first.



Warning: You can upgrade to Cloudera Manager 6.0 from Cloudera Manager/CDH versions 5.7 through 5.14, but **NOT** from Cloudera Manager/CDH 5.15.x or 5.16.x. You will be able to upgrade to a future version of Cloudera Manager 6.x from version 5.15.x and 5.16.x.



Warning: Downtime for the cluster is **not required** to complete the Cloudera Manager upgrade. However, a rolling restart to the CDH cluster is **recommended** and can be scheduled independently of the Cloudera Manager upgrade. Rolling upgrade of CDH 5 to CDH 6 is not supported, so downtime will be **required** when upgrading the CDH cluster.



Warning: Cloudera Data Science Workbench is not supported with Cloudera Manager 6.0.x. If you proceed with the upgrade, you will be asked to remove the CDSW service from your cluster. Cloudera Data Science Workbench 1.5 (and higher) is supported with Cloudera Manager 6.1 (and higher).



Important: Cloudera Manager 6.x only supports Redhat/Centos 6 or 7.



Important: Cloudera Manager 6.x only supports SLES 12.



Important: Cloudera Manager 6.x only supports Ubuntu 16.04 (Xenial).



Important: Cloudera Manager 6.x does not support Debian.



Important: Oracle 11 is not supported in Cloudera Manager 6.x. If your deployment uses Oracle 11 as the database for Cloudera Manager, you must upgrade to a [supported database](#) before upgrading Cloudera Manager.

Collect Information

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Collect the following information about your environment and fill in the form above. This information will be remembered by your browser on all pages in this Upgrade Guide.

- a. The current version of the Operating System:

```
lsb_release -a
```

Database parameters:

```
cat /etc/cloudera-scm-server/db.properties
```

```
...
com.cloudera.cmf.db.type=mysql
com.cloudera.cmf.db.host=database_hostname:database_port
com.cloudera.cmf.db.name=scm
com.cloudera.cmf.db.user=scm
com.cloudera.cmf.db.password=SOME_PASSWORD
```

- b. Log in to the Cloudera Manager Admin console and find the following:

- a. The version of Cloudera Manager used in your cluster. Go to **Support > About**.
- b. The version of the JDK deployed in the cluster. Go to **Support > About**.

Preparing to Upgrade Cloudera Manager

- You must have SSH access to the Cloudera Manager server hosts and be able to log in using the root account or an account that has password-less sudo permission for all hosts.
- Review the following when upgrading to Cloudera Manager 5.x:
 - [Requirements and Supported Versions](#)
 - [Operating System Requirements](#)
 - [Database Requirements](#)
 - [Java Requirements](#)
- Review the following when upgrading to Cloudera Manager 6.x:
 - [Cloudera Enterprise 6 Requirements and Supported Versions](#)
 - [Operating System Requirements](#)
 - [Database Requirements](#)
 - [Java Requirements](#)
- You should have already upgraded to a supported operating system before upgrading Cloudera Manager. See [Upgrading the Operating System](#) on page 29.
- You should have either Oracle JDK 1.7 or JDK 1.8 (recommended) already installed on all the hosts or have it installed on the Cloudera Manager server host and select the **Install Oracle Java SE Development Kit** checkbox in the Upgrade Cloudera Manager Agent Packages wizard. See [Upgrading the JDK](#) on page 22.

Cloudera Manager 6.0.0 and higher and CDH 6.0.0 and higher require Oracle JDK 8.

If you plan to use Spark 2.2 using Cloudera Manager 5.x, you must install Oracle JDK 1.8 manually and configure the **Java Home Directory** property for all the hosts.

- Review the Release Notes:
 - [Cloudera Enterprise 6 Release Guide](#)
 - [Cloudera Enterprise 5.x Release Notes](#)
- Review the [Cloudera Security Bulletins](#).
- Downtime for the cluster is not required to complete the Cloudera Manager upgrade.
- Downtime for the cluster is not required to complete the Cloudera Manager upgrade; for services that contribute to Cloudera Navigator audits, lineage, or search, **restarting the service is recommended** but can be scheduled independently of the upgrade.
- The embedded PostgreSQL database is **not supported** in production environment because downtime is **required**.
- If you have previously installed Kafka 1.2, and are upgrading from Cloudera Manager 5.4 or lower, remove the Kafka CSD:
 1. Determine the location of the CSD directory:
 - a. Select **Administration > Settings**.
 - b. Click the **Custom Service Descriptors** category.
 - c. Retrieve the directory from the **Local Descriptor Repository Path** property.
 2. Delete the Kafka CSD from the directory.
- [Upgrade Cloudera Manager using Tarballs is only supported for upgrades to 5.x and is now deprecated.](#)
- If your cluster uses Oracle for any databases, before upgrading from CDH 5 to CDH 6, check the value of the COMPATIBLE initialization parameter in the Oracle Database using the following SQL query:

```
SELECT name, value FROM v$parameter WHERE name = 'compatible'
```

The default value is 12.2.0. If the parameter has a different value, you can set it to the default as shown in the [Oracle Database Upgrade Guide](#).



Note: Before resetting the COMPATIBLE initialization parameter to its default value, make sure you consider the effects of this change can have on your system.

Backing Up Cloudera Manager

This topic contains procedures to back up Cloudera Manager. Cloudera recommends that you perform these backup steps before upgrading. The backups will allow you to rollback your Cloudera Manager upgrade if needed.



Warning: The selected version of Cloudera Manager is too old and cannot be upgraded to 6.x. Upgrade your deployment to Cloudera Manager 5.7 or higher first.



Warning: You can upgrade to Cloudera Manager 6.0 from Cloudera Manager/CDH versions 5.7 through 5.14, but **NOT** from Cloudera Manager/CDH 5.15.x or 5.16.x. You will be able to upgrade to a future version of Cloudera Manager 6.x from version 5.15.x and 5.16.x.

Collect Information for Backing Up Cloudera Manager

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Collect database information by running the following command:

```
cat /etc/cloudera-scm-server/db.properties
```

For example:

```
...
com.cloudera.cmf.db.type=...
com.cloudera.cmf.db.host=database_hostname:database_port
com.cloudera.cmf.db.name=scm
com.cloudera.cmf.db.user=scm
com.cloudera.cmf.db.password=SOME_PASSWORD
```

3. Collect information (host name, port number, database name, user name and password) for the following databases.

- Reports Manager
- Navigator Audit Server
- Navigator Metadata Server
- Activity Monitor

You can find the database information by using the Cloudera Manager Admin Console. Go to **Clusters > Cloudera Management Service > Configuration** and select the **Database** category. You may need to contact your database administrator to obtain the passwords.

4. Find the host where the Service Monitor, Host Monitor and Event Server roles are running. Go to **Clusters > Cloudera Manager Management Service > Instances** and note which hosts are running these roles.
5. Identify the location of the Cloudera Navigator Metadata Server storage directory:
 - a. Go to **Clusters > Cloudera Management Service > Instances**.
 - b. Click the **Configuration** tab.
 - c. Select **Scope > Navigator Metadata Server**.
 - d. The **Navigator Metadata Server Storage Dir** property stores the location of the directory.
6. Ensure that Navigator Metadata Server Java heap is large enough to complete the upgrade. You can estimate the amount of heap needed from the number of elements and relations stored in the Solr storage directory.
 - a. Go to **Clusters > Cloudera Management Service > Instances**.
 - b. In the list of instances, click **Navigator Metadata Server**.
 - c. Select **Log Files > Role Log File**.
 - d. Search the log file for `solr core nav_elements` and note the number of element documents.
 - e. Search the log file for `solr core nav_relations` and note the number of relation documents.
 - f. Multiply the total number of documents by 200 bytes per document and add to it a baseline of 2 GB:

```
((num_nav_elements + num_nav_relations) * 200 bytes) + 2 GB
```

For example, if you had 68813088 elements and 78813930 relations, the recommended Java heap size is ~30 GB:

```
((68813088 + 78813930) * 200) + 2 GB = 29525403600 bytes = ~29.5 GB + 2 GB = ~ 31.5 GB
```

- g. Set the heap value in the Java Heap Size of Navigator Metadata Server in Bytes property in **Clusters > Cloudera Management Service > Configuration**.

Back Up Cloudera Manager Agent



Note: Commands are provided below to backup various files and directories used by Cloudera Manager Agents. If you have configured custom paths for any of these, substitute those paths in the commands. The commands also provide destination paths to store the backups, defined by the environment variable `CM_BACKUP_DIR`, which is used in all the backup commands. You may change these destination paths in the command as needed for your deployment.

The `tar` commands in the steps below may return the following message. It is safe to ignore this message:

```
tar: Removing leading `/' from member names
```

Backup up the following Cloudera Manager agent files on **all hosts**:

- Create a top level backup directory.

```
export CM_BACKUP_DIR=`date +%F`-CM
echo $CM_BACKUP_DIR
mkdir -p $CM_BACKUP_DIR
```

- Back up the Agent directory and the runtime state.

```
sudo -E tar -cf $CM_BACKUP_DIR/cloudera-scm-agent.tar --exclude=*.sock
/etc/cloudera-scm-agent /etc/default/cloudera-scm-agent /var/run/cloudera-scm-agent
/var/lib/cloudera-scm-agent
```

- Back up the existing repository directory.

RHEL / CentOS

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/yum.repos.d
```

SLES

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/zypp/repos.d
```

Debian / Ubuntu

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/apt/sources.list.d
```

Back Up the Cloudera Management Service



Note: Commands are provided below to backup various files and directories used by Cloudera Manager Agents. If you have configured custom paths for any of these, substitute those paths in the commands. The commands also provide destination paths to store the backups. You may change these destination paths in the command as needed for your deployment.

1. On the host where the Service Monitor role is configured to run, backup the following directory:

```
sudo cp -rp /var/lib/cloudera-service-monitor /var/lib/cloudera-service-monitor-`date
+%F`-CM
```

2. On the host where the Host Monitor role is configured to run, backup the following directory:

```
sudo cp -rp /var/lib/cloudera-host-monitor /var/lib/cloudera-host-monitor-`date +%F`-CM
```

3. On the host where the Event Server role is configured to run, back up the following directory:

```
sudo cp -rp /var/lib/cloudera-scm-eventserver /var/lib/cloudera-scm-eventserver-`date +%F`-CM
```

Back Up Cloudera Navigator Data

- 1.



Important: Upgrading from Cloudera Manager 5.9 (Navigator 2.8) and earlier can take a significant amount of time, depending on the size of the Navigator Metadata storage directory. When the Cloudera Manager upgrade process completes and Cloudera Navigator services restart, the Solr indexing upgrade automatically begins. No other actions can be performed until Solr indexing completes (a progress message displays during this process). It can take as long as two days to upgrade a storage directory with 60 GB. To help mitigate this extended upgrade step, make sure to clear out all unnecessary metadata using purge, check the size of the storage directory, and consider rerunning purge with tighter conditions to further reduce the size of the storage directory.

2. Make sure a purge task has run recently to clear stale and deleted entities.

- You can see when the last purge tasks were run in the Cloudera Navigator console (From the Cloudera Manager Admin console, go to **Clusters > Cloudera Navigator**. Select **Administration > Purge Settings**.)
- If a purge hasn't run recently, run it by editing the Purge schedule on the same page.
- Set the purge process options to clear out as much of the backlog of data as you can tolerate for your upgraded system. See [Managing Metadata Storage with Purge](#).

3. Stop the Navigator Metadata Server.

- a. Go to **Clusters > Cloudera Management Service > Instances**.
- b. Select **Navigator Metadata Server**.
- c. Click **Actions for Selected > Stop**.

4. Back up the Cloudera Navigator Solr storage directory.

```
sudo cp -rp /var/lib/cloudera-scm-navigator /var/lib/cloudera-scm-navigator-`date +%F`-CM
```

5. If you are using an Oracle database for audit, in SQL*Plus, ensure that the following additional privileges are set:

```
GRANT EXECUTE ON sys.dbms_crypto TO nav;  
GRANT CREATE VIEW TO nav;
```

where *nav* is the user of the Navigator Audit Server database.

Stop Cloudera Manager Server & Cloudera Management Service

1. Stop the **Cloudera Management Service**.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters > Cloudera Management Service**.
- c. Select **Actions > Stop**.

2. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

3. Stop the Cloudera Manager Server.**RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher**

```
sudo systemctl stop cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server stop
```

Back Up the Cloudera Manager Databases

1. Back up the **Cloudera Manager server database** – Run the following command. (The command displayed below depends on the database you selected in the form at the top of this page. Replace placeholders with the actual values returned from the `db.properties` file):

MySQL

```
mysqldump --databases database_name --host=database_hostname --port=database_port -u user_name -p > $HOME/database_name-backup-`date +%F`-CM.sql
```



Note: If the `db.properties` file does not contain a port number, omit the port number parameter from the above command.

PostgreSQL/Embedded

```
pg_dump -h database_hostname -U user_name -W -p database_port database_name > $HOME/database_name-backup-`date +%F`-CM.sql
```

Oracle

Work with your database administrator to ensure databases are properly backed up.

For more information about backing up databases, see [Backing Up Databases](#).

2. Back up **All other Cloudera Manager databases** - Use the database information that [you collected in a previous step](#). You may need to contact your database administrator to obtain the passwords.

These databases can include the following:

- Reports Manager
- Navigator Audit Server
- Navigator Metadata Server
- Activity Monitor (Only used for MapReduce 1 monitoring).

Run the following commands to back up the databases. (The command displayed below depends on the database you selected in the form at the top of this page. Replace placeholders with the actual values.):

MySQL

```
mysqldump --databases database_name --host=database_hostname --port=database_port -u database_username -p > $HOME/database_name-backup-`date +%F`-CM.sql
```

PostgreSQL/Embedded

```
pg_dump -h database_hostname -U database_username -W -p database_port database_name > $HOME/database_name-backup-`date +%F`-CM.sql
```

Oracle

Work with your database administrator to ensure databases are properly backed up.

Back Up Cloudera Manager Server



Note: Commands are provided below to backup various files and directories used by Cloudera Manager Agents. If you have configured custom paths for any of these, substitute those paths in the commands. The commands also provide destination paths to store the backups, defined by the environment variable `CM_BACKUP_DIR`, which is used in all the backup commands. You may change these destination paths in the command as needed for your deployment.

The `tar` commands in the steps below may return the following message. It is safe to ignore this message:

```
tar: Removing leading `/' from member names
```

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Create a top-level backup directory.

```
export CM_BACKUP_DIR=`date +%F`-CM
echo $CM_BACKUP_DIR
mkdir -p $CM_BACKUP_DIR
```

3. Back up the Cloudera Manager Server directories:

```
sudo -E tar -cf $CM_BACKUP_DIR/cloudera-scm-server.tar /etc/cloudera-scm-server
/etc/default/cloudera-scm-server
```

4. Back up the existing repository directory.

RHEL / CentOS

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/yum.repos.d
```

SLES

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/zypp/repos.d
```

Debian / Ubuntu

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/apt/sources.list.d
```

(Optional) Start Cloudera Manager Server & Cloudera Management Service

Start the Cloudera Manager server and Cloudera Manager Management service.

If you will be immediately upgrading Cloudera Manager, skip this step and continue with [Upgrading the Cloudera Manager Server](#) on page 47.

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Start the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server
```

If the Cloudera Manager Server starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server start
```

You should see the following:

```
Starting cloudera-scm-server: [ OK ]
```

3. Start the **Cloudera Management Service**.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters > Cloudera Management Service**.
- c. Select **Actions > Start**.

Upgrading the Cloudera Manager Server

This topic provides procedures for backing up the Cloudera Manager Server.

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)



Warning: Redhat 5 is not a supported operating system for Cloudera Manager 6.x.



Warning: SLES 11 is not a supported operating system for Cloudera Manager 6.x.



Warning: Ubuntu 14 and lower are not supported operating systems for Cloudera Manager 6.x.



Warning: Debian is not a supported operating system for Cloudera Manager 6.x.



Warning: Ubuntu 12 is not a supported operating systems for Cloudera Manager 5.16 and higher.



Warning: The selected version of Cloudera Manager is too old and cannot be upgraded to 6.x. Upgrade your deployment to Cloudera Manager 5.7 or higher first.



Warning: You can upgrade to Cloudera Manager 6.0 from Cloudera Manager/CDH versions 5.7 through 5.14, but **NOT** from Cloudera Manager/CDH 5.15.x or 5.16.x. You will be able to upgrade to a future version of Cloudera Manager 6.x from version 5.15.x and 5.16.x.

After you complete the steps in [Getting Started Upgrading Cloudera Manager](#) on page 38 and [Backing Up Cloudera Manager](#) on page 41, continue with the following:



Important: If you encounter problems, see the following:

- [Troubleshooting a Cloudera Manager Upgrade](#) on page 62
- [Reverting a Failed Cloudera Manager Upgrade](#) on page 63

Establish Access to the Software

Cloudera Manager needs access to a package repository that contains the updated software packages.

- If the Cloudera Manager hosts have internet access, you can use the publicly available repositories from <https://archive.cloudera.com>. Modify the sample repo files below for your operating system and version.
- If the Cloudera Manager hosts *do not* have internet access, configure [a local package repository](#) hosted on your network. For example: `http://MyWebServer:1234/cloudera-repos`
- Replace `archive.cloudera.com` in the sample repo files below with the URL for your local repository.

Package Repository URL:

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Remove any older files in the existing repository directory:

RHEL / CentOS

```
sudo rm /etc/yum.repos.d/cloudera*manager.repo*
```

SLES

```
sudo rm /etc/zypp/repos.d/cloudera*manager.repo*
```

Debian / Ubuntu

```
sudo rm /etc/apt/sources.list.d/cloudera*.list*
```

3. Fill in the form at the top of this page.
4. Create a repository file so that the package manager can locate and download the binaries:

RHEL / CentOS

Create a file named `/etc/yum.repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
baseurl=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```


SLES

Create a file named `/etc/zypp/repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
baseurl=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

Debian / Ubuntu

Create a file named `/etc/apt/sources.list.d/cloudera_manager.list` with the following content:

```
# Packages for Cloudera Manager
deb https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
deb-src https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
```

```
sudo apt-get update
```

The repository file, as created, refers to the *most recent* maintenance release of the specified minor release. If you would like to use a *specific* maintenance version, for example 5.15.1, replace 5.15 with 5.15.1 in the generated repository file shown above.

5. A Cloudera Manager upgrade can introduce new package dependencies. Your organization may have restrictions or require prior approval for installation of new packages. You can determine which packages may be installed or upgraded:

RHEL / CentOS

```
yum deplist cloudera-manager-agent
```

SLES

```
zypper info --requires cloudera-manager-agent
```

Debian / Ubuntu

```
apt-cache depend cloudera-manager-agent
```

Install JDK 8

Oracle JDK 1.8 is required on all cluster hosts managed by Cloudera Manager 6.0.0 or higher. If JDK 1.8 is already installed on your hosts, skip the steps in this section.

If you are upgrading to Cloudera Manager 6.0.0 or higher, you can manually install JDK 8 on the Cloudera Manager server host, and then, as part of the Cloudera Manager upgrade process, you can specify that Cloudera Manager upgrade the JDK on the remaining hosts.

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Install JDK 1.8.

RHEL / CentOS

```
sudo yum install oracle-j2sdk1.8.x86_64
```

SLES

```
sudo zypper install oracle-j2sdk1.8.x86_64
```

Debian / Ubuntu

```
sudo apt-get install oracle-j2sdk1.8
```

3. Open the following file in a text editor:

```
/etc/default/cloudera-scm-server
```

4. Edit the line that begins with `export JAVA_HOME` (if this line does not exist, add it) and change the path to the path of the new JDK (you can find the path under `/usr/java`).

For example: (RHEL and SLES)

```
export JAVA_HOME="/usr/java/jdk1.8.0_141-cloudera"
```

For example: (Ubuntu)

```
export JAVA_HOME="/usr/lib/jvm/java-8-oracle-cloudera"
```

5. Save the file.

Upgrade the Cloudera Manager Server

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. If your cluster is running the embedded PostgreSQL database, stop all services that are using the embedded database. These can include:

- Hive service and all services such as Impala and Hue that use the Hive metastore
- Oozie
- Sentry
- Sqoop

3. Stop the **Cloudera Management Service**.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters > Cloudera Management Service**.
- c. Select **Actions > Stop**.



Important: Not stopping the Cloudera Management Service at this point might cause management roles to crash or the Cloudera Manager Server might fail to restart.

4. Ensure that you have disabled any scheduled **replication or snapshot jobs** and wait for any **running commands** from the Cloudera Manager Admin Console to complete before proceeding with the upgrade.



Important: If there are replication jobs, snapshot jobs, or other commands running when you stop Cloudera Manager Server, Cloudera Manager Server might fail to start after the upgrade.

5. Stop the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server stop
```

6. If you are using the embedded PostgreSQL database, stop the Cloudera Manager Embedded PostgreSQL database:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server-db
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server-db stop
```

If you are not using the embedded PostgreSQL database and you attempt to stop it, you might see a message indicating that the service cannot be found. If you see a message that the shutdown failed, then the embedded database is still running, probably because services are connected to the Hive metastore. If the database shutdown fails due to connected services, issue the following command:

RHEL-compatible 7 and higher, Ubuntu 16.04

```
sudo service cloudera-scm-server-db next_stop_fast
sudo service cloudera-scm-server-db stop
```

All other Linux distributions

```
sudo service cloudera-scm-server-db fast_stop
```

7. Stop the **Cloudera Manager Agent**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-agent
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent stop
```

8. Upgrade the packages. Include the `cloudera-manager-server-db-2` package in the command only if you are using the embedded PostgreSQL database.

RHEL / CentOS

```
sudo yum clean all
sudo yum upgrade cloudera-manager-server cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

SLES

```
sudo zypper clean --all
sudo zypper up cloudera-manager-server cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

Debian / Ubuntu

```
sudo apt-get clean
```

```
sudo apt-get update
```

```
sudo apt-get dist-upgrade
```

```
sudo apt-get install cloudera-manager-server cloudera-manager-daemons  
cloudera-manager-agent cloudera-manager-server-db-2
```

You might be prompted about your configuration file version:

```
Configuration file '/etc/cloudera-scm-agent/config.ini'  
==> Modified (by you or by a script) since installation.  
==> Package distributor has shipped an updated version.  
What would you like to do about it ? Your options are:  
Y or I : install the package maintainer's version  
N or O : keep your currently-installed version  
D : show the differences between the versions  
Z : start a shell to examine the situation  
The default action is to keep your current version.
```

You may receive a similar prompt for `/etc/cloudera-scm-server/db.properties`. Answer **N** to both prompts.

You may be prompted to accept the GPG key. Answer **y**.

```
Retrieving key from https://archive.cloudera.com/.../cm/RPM-GPG-KEY-cloudera  
Importing GPG key ...  
Userid : "Yum Maintainer <webmaster@cloudera.com>"  
Fingerprint: ...  
From : https://archive.cloudera.com/.../RPM-GPG-KEY-cloudera
```



Note: If you receive the following error message when running these commands: **[Errno 14] HTTP Error 404 - Not Found**, make sure the URL in the `cloudera-manage.list` `cloudera-manager.repo` file is correct and is reachable from the Cloudera Manager server host.

9. If you customized the `/etc/cloudera-scm-agent/config.ini` file, your customized file is renamed with the extension `.rpmsave` or `.dpkg-old`. Merge any customizations into the `/etc/cloudera-scm-agent/config.ini` file that is installed by the package manager.
10. Verify that you have the correct packages installed.

Debian / Ubuntu

```
dpkg-query -l 'cloudera-manager-*
```

```
Desired=Unknown/Install/Remove/Purge/Hold  
| Status=Not/Inst/Conf-files/Unpacked/halfF-conf/Half-inst/trig-aWait/Trig-pend  
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)  
||/ Name Version Description  
++-----+-----+-----+  
ii cloudera-manager-agent 5.15.0-0.cm...~sq The Cloudera Manager Agent
```

```
ii cloudera-manager-daemo 5.15.0-0.cm...~sq Provides daemons for monitoring Hadoop and
related tools.
ii cloudera-manager-serve 5.15.0-0.cm...~sq The Cloudera Manager Server
```

RHEL / CentOS / SLES

```
rpm -qa 'cloudera-manager-*
```

```
cloudera-manager-server-5.15.0-..cm...
cloudera-manager-agent-5.15.0-..cm...
cloudera-manager-daemons-5.15.0-..cm...
cloudera-manager-server-db-2-5.15.0-..cm...
```

11 If you are using the embedded PostgreSQL database, start the database:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server-db
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server-db start
```

12 Start the Cloudera Manager Agent.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-agent
```

If the agent starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent start
```

You should see the following:

```
Starting cloudera-scm-agent: [ OK ]
```

13 Start the Cloudera Manager Server.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server
```

If the Cloudera Manager Server starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server start
```

You should see the following:

```
Starting cloudera-scm-server: [ OK ]
```

14 Use a Web browser to open the Cloudera Manager Admin Console using the following URL:

Cloudera Manager 5.15 and higher:

```
http://cloudera_Manager_server_hostname:7180/cm/upgrade
```

Upgrading Cloudera Manager

Cloudera Manager 5.14 and lower:

```
http://cloudera_manager_server_hostname:7180/cmf/upgrade-wizard/welcome
```

It can take several minutes for the Cloudera Manager Server to start, and the Cloudera Manager Admin Console is unavailable until the server startup is complete and the **Upgrade Cloudera Manager** page displays. Continue with the steps on the next page to upgrade the Cloudera Manager Agents.



Note: If you have problems starting the server or the agent, such as database permissions problems, you can use log files to troubleshoot the problem:

Server log:

```
tail -f /var/log/cloudera-scm-server/cloudera-scm-server.log
```

Agent log:

```
tail -f /var/log/cloudera-scm-agent/cloudera-scm-agent.log
```

or

```
tail -f /var/log/messages
```

To complete the Cloudera Manager upgrade, continue with [Upgrading the Cloudera Manager Agents](#) on page 54.

Upgrading the Cloudera Manager Agents

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)



Warning: Redhat 5 is not a supported operating system for Cloudera Manager 6.x.



Warning: SLES 11 is not a supported operating system for Cloudera Manager 6.x.



Warning: Ubuntu 14 and lower are not supported operating systems for Cloudera Manager 6.x.



Warning: Debian is not a supported operating system for Cloudera Manager 6.x.



Warning: Ubuntu 12 is not a supported operating systems for Cloudera Manager 5.16 and higher.



Warning: The selected version of Cloudera Manager is too old and cannot be upgraded to 6.x. Upgrade your deployment to Cloudera Manager 5.7 or higher first.



Warning: You can upgrade to Cloudera Manager 6.0 from Cloudera Manager/CDH versions 5.7 through 5.14, but **NOT** from Cloudera Manager/CDH 5.15.x or 5.16.x. You will be able to upgrade to a future version of Cloudera Manager 6.x from version 5.15.x and 5.16.x.



Important: If you encounter problems, see the following:

- [Troubleshooting a Cloudera Manager Upgrade](#) on page 62

Upgrade the Cloudera Manager Agents

You can upgrade the agents using one of the two options below.

Option 1. Upgrade the Agents using Cloudera Manager (Recommended)



Note: (SLES 12 only) If Cloudera Manager displays an error message during this procedure that is similar to the following, you may have an old version of the JDK that is interfering with the upgrade:

```
ls: cannot access '/etc/zypp/repos.d/cloudera-manager.repo.*': No such
file or directory
repository file /etc/zypp/repos.d/cloudera-manager.repo removed
```

Removing the old JDK before upgrading will allow Cloudera Manager to complete the upgrade.

After the Cloudera Manager Server starts and you log in to the Cloudera Manager Admin Console, the Upgrade Cloudera Manager page displays. (It might take several minutes for the server to start.)

If the Upgrade Cloudera Manager page does not display after you upgraded the packages on the Cloudera Manager Server host, open the following URL in your web browser:

Cloudera Manager 5.15 or higher:

```
https://my_cloudera_manager_server_host:port/cm/upgrade
```

Cloudera Manager 5.14 or lower:

```
https://my_cloudera_manager_server_host:port/cm/upgrade-wizard/welcome
```

The status of Agent upgrades is displayed in one or more groups. Because you might have only upgraded the Cloudera Manager Agent on the Cloudera Manager Server host, the first group shows that host as having an upgraded agent. If the hosts managed by Cloudera Manager have different operating systems, a group for each operating system displays the Agent upgrade status for those hosts.

Follow the instructions on the upgrade page to upgrade all the agents.

1. Cloudera Manager 5.14 or lower: Click **Yes, I would like to upgrade Cloudera Manager Agent packages now.**
2. Click **Continue.**

The **Upgrade Cloudera Manager Agent Packages** page displays.

3. Cloudera Manager 5.15 or higher: If there are more than one group of hosts that require agent upgrades, select the group from the drop-down list labeled **Upgrade Cloudera Manager Agent Packages running on:**. If there is only one group that requires upgrades, this drop-down list does not appear.

In Cloudera Manager 5.15 or higher, Cloudera Manager can upgrade agents even when they are running on different operating systems or versions (one OS group at a time).

4. Click **Upgrade Cloudera Manager Agent packages.**

The **Upgrade Cloudera Manager Agent Packages** page displays.

5. If you are using a local package repository instead of the public repository at <https://archive.cloudera.com>, select the **Custom Repository** option when installing the Cloudera Manager Agent packages and enter the **Custom Repository URL**.

In Cloudera Manager 5.x:

RHEL / CentOS

Use the `baseurl` value in the `cloudera-manager.repo` file as the **Custom Repository**. Use the `gpgkey` value as the **Custom GPG Key URL**.

SLES

Use the `baseurl` value in the `cloudera-manager.repo` file as the **Custom Repository**. Use the `gpgkey` value as the **Custom GPG Key URL**.

Debian / Ubuntu

Use the entire `deb url contrib` line from the `cloudera_Manager.list` file as the **Custom Repository**. Use the `url/archive.key` as the **Custom GPG Key URL**.

In Cloudera Manager 6 or higher:

Use the `Package Repository URL/cm6/6.0.x` as the **Custom Repository**.

6. Click **Continue**.

The **Accept JDK License** page displays.

7. If you want to install JDK on all hosts, select **Install Oracle Java SE Development Kit**.

8. Click **Continue**.

The **Enter Login Credentials** page displays.

9. Specify the credentials and initiate Agent installation:

- a. Select **root** for the `root` account, or select **Another user** and enter the username for an account that has password-less `sudo` permission.
- b. Select an authentication method:
 - If you choose the **All hosts accept same password** option, enter and confirm the password.
 - If you choose the **All hosts accept same private key** option, provide a passphrase and path to the required key files.
- c. Modify the default SSH port if necessary.
- d. Specify the maximum **Number of Simultaneous Installations** to run at once. The default and recommended value is 10. Adjust this parameter based on your network capacity.

10. Click **Continue**.

The Cloudera Manager Agent packages and, if selected, the JDK are installed.

11. When the installations complete, click **Finish**.

The **Upgrade Cloudera Manager** page displays the status of the upgrade.

If there are additional groups of hosts that require Agent upgrades, select the next group from the **Upgrade Cloudera Manager Agent Packages running on:** drop-down list, and repeat the agent installation steps.

12. Click **Run Host Inspector** to run the host inspector. Inspect the output and correct any warnings. If problems occur, you can make changes and then rerun the inspector.
13. When you are satisfied with the inspection results, start the Cloudera Management Service.
14. Click the link at the bottom of the page to go back to the **Home Page**.
15. Click **Continue**.

The Cloudera Manager Agent packages and, if selected, the JDK are installed.

The Host Inspector runs to inspect your managed hosts for correct versions and configurations. If problems occur, you can make changes and then click **Run Again** to rerun the inspector.

16 Click Continue.

The **Review Changes** page displays and suggests configuration changes you may need to make.

17. Make any necessary changes and click Continue.

The Upgrade Wizard restarts the **Cloudera Manager Management Service**.

18 Click Finish.

19 The Cloudera Manager Home page opens and displays the status of the cluster. It can take several minutes for all of the services to display their current status. You may need to restart some services or redeploy stale client configurations.

Option 2. Upgrade the Agents using the Command Line

- If the Cloudera Manager hosts have internet access, you can use the publicly available repositories from <https://archive.cloudera.com>. Modify the sample repo files below for your operating system and version.
- If the Cloudera Manager hosts *do not* have internet access, configure [a local package repository](#) hosted on your network. For example: `http://MyWebServer:1234/cloudera-repos`
- Replace `archive.cloudera.com` in the sample repo files below with the URL for your local repository.

Perform the following commands on all hosts managed by Cloudera Manager:

(You can also multiplex the same set of commands to all hosts by using utilities such as `csshX`, `pdsh`, or `pssh`.)

1. Log in to each host using ssh. For example:

```
ssh host1.example.com
```

2. Remove any older files in the existing repository directory:

RHEL / CentOS

```
sudo rm /etc/yum.repos.d/cloudera*manager.repo*
```

SLES

```
sudo rm /etc/zypp/repos.d/cloudera*manager.repo*
```

Debian / Ubuntu

```
sudo rm /etc/apt/sources.list.d/cloudera*.list*
```

3. Tip: If you have a mixed operating system environment, adjust the **Operating System** filter at the top of the page for each operating system. The guide will generate the repo file for you automatically here.

RHEL / CentOS

Create a file named `/etc/yum.repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
baseurl=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

SLES

Create a file named `/etc/zypp/repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
baseurl=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

Debian / Ubuntu

Create a file named `/etc/apt/sources.list.d/cloudera_manager.list` with the following content:

```
# Packages for Cloudera Manager
deb https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
deb-src https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
```

```
sudo apt-get update
```

4. Stop the Cloudera Manager Agent.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-agent
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent stop
```

5. Install JDK 1.8.

RHEL / CentOS

```
sudo yum install oracle-j2sdk1.8.x86_64
```

SLES

```
sudo zypper install oracle-j2sdk1.8.x86_64
```

Debian / Ubuntu

```
sudo apt-get install oracle-j2sdk1.8
```

6. Upgrade the agent packages.

Note: Only add `cloudera-manager-server-db-2` if you are using the embedded PostgreSQL database.

RHEL / CentOS

```
sudo yum clean all
sudo yum repolist
```

```
sudo yum upgrade cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

SLES

```
sudo zypper clean --all
```

```
sudo zypper up cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

Debian / Ubuntu

```
sudo apt-get clean
sudo apt-get update
sudo apt-get dist-upgrade
```

```
sudo apt-get install cloudera-manager-daemons cloudera-manager-agent
cloudera-manager-server-db-2
```

You might be prompted about your configuration file version:

```
Configuration file '/etc/cloudera-scm-agent/config.ini'
==> Modified (by you or by a script) since installation.
==> Package distributor has shipped an updated version.
What would you like to do about it ? Your options are:
Y or I : install the package maintainer's version
N or O : keep your currently-installed version
D : show the differences between the versions
Z : start a shell to examine the situation
The default action is to keep your current version.
```

You may receive a similar prompt for `/etc/cloudera-scm-server/db.properties`. Answer **N** to both prompts.

7. If you customized the `/etc/cloudera-scm-agent/config.ini` file, your customized file is renamed with the extension `.rpmsave` or `.dpkg-old`. Merge any customizations into the `/etc/cloudera-scm-agent/config.ini` file that is installed by the package manager.
8. Verify that you have the correct packages installed.

Debian / Ubuntu

```
dpkg-query -l 'cloudera-manager-*'
```

```
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/half-f-inst/Trig-await/Trig-pend
| Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name Version Description
+++=-----+-----+-----+
ii cloudera-manager-agent 5.15.0-0.cm...~sq The Cloudera Manager Agent
ii cloudera-manager-daemo 5.15.0-0.cm...~sq Provides daemons for monitoring Hadoop and
related tools.
```

RHEL / CentOS / SLES

```
rpm -qa 'cloudera-manager-*'
```

```
cloudera-manager-agent-5.15.0-..cm...
cloudera-manager-daemons-5.15.0-..cm...
cloudera-manager-server-db-2-5.15.0-..cm...
```

9. If you are using the embedded PostgreSQL database, start the database:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server-db
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server-db start
```

- 10 Start the Cloudera Manager Agent.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-agent
```

If the agent starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent start
```

You should see the following:

```
Starting cloudera-scm-agent: [ OK ]
```

- 11 In Cloudera Manager 5.15 or higher, you can monitor the progress at https://my_cloudera_manager_server_host:port/cm/upgrade.
- 12 When all the agents are upgraded, run the **Host Inspector**.
- 13 In Cloudera Manager 5.14 or lower, Run the **Run Host Inspector**: Select **Hosts > All Hosts** and click **Inspect All Hosts**.
- 14 The Host Inspector runs to inspect your managed hosts for correct versions and configurations. If problems occur, you can make changes and then rerun the inspector.
- 15 The Cloudera Manager Home page opens and displays the status of the cluster. It can take several minutes for all of the services to display their current status. You may need to restart some services or redeploy stale client configurations.

If you are upgrading from Cloudera Manager 5.5.0 or lower to Cloudera Manager 5.5.0 or higher, hard restart the agent on all hosts to update and restart the `supervisord` process.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo sudo systemctl stop supervisord
sudo systemctl start cloudera-scm-agent
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent hard_restart
```

After You Upgrade Cloudera Manager

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)



Warning: The selected version of Cloudera Manager is too old and cannot be upgraded to 6.x. Upgrade your deployment to Cloudera Manager 5.7 or higher first.



Warning: You can upgrade to Cloudera Manager 6.0 from Cloudera Manager/CDH versions 5.7 through 5.14, but **NOT** from Cloudera Manager/CDH 5.15.x or 5.16.x. You will be able to upgrade to a future version of Cloudera Manager 6.x from version 5.15.x and 5.16.x.

Upgrade Cloudera Navigator Encryption Components

Upgrade any Cloudera Navigator Encryption components deployed in your cluster:

- Cloudera Navigator Key Trustee Server
- Cloudera Navigator Key HSM
- Cloudera Navigator Key Trustee KMS
- Cloudera Navigator Encrypt.

If you are still using **Key Trustee Server** 5.4, and you are upgrading to Cloudera Manager 5.10 or higher, you must upgrade the Key Trustee Server to a more recent version.

You can upgrade other Cloudera Navigator components at any time. You do not have to perform these upgrades when upgrading Cloudera Manager or CDH.

See [Upgrading Cloudera Navigator Data Encryption](#) on page 152.

Perform Post Upgrade Steps

1. If you upgraded the JDK, do the following:

- If the Cloudera Manager Server host is also running a Cloudera Manager Agent, restart the Cloudera Manager Server:
- Restart the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl restart cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

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

c. Open the Cloudera Manager Admin Console and set the **Java Home Directory** property in the host configuration:

- Go to **Home > All Hosts > Configuration**.
- Set the value to the path to the new JDK.
- Click **Save Changes**.

d. Restart all services:

- On the **Home > Status** tab, click



next to the cluster name, select **Restart** and confirm.

2. Start the **Cloudera Management Service** and adjust any configurations when prompted.

- Log in to the Cloudera Manager Admin Console.
- Select **Clusters > Cloudera Management Service**.
- Select **Actions > Start**.

3. If Cloudera Manager reports [stale configurations](#) after the upgrade, you might need to restart the cluster services and redeploy the client configurations. If you are also upgrading CDH, this step is not required.

Stale configurations can occur after a Cloudera Manager upgrade when a default configuration value has changed, which is often required to fix a serious problem. Configuration changes that result in Cloudera Manager reporting stale configurations are described the release notes:

- [Cloudera Enterprise 6 Release Guide](#)
- [Cloudera Enterprise 5.x Release Notes](#)

a. On the **Home > Status** tab, click



next to the cluster name, select **Restart** and confirm.

b. On the **Home > Status** tab, click



next to the cluster name, select **Deploy Client Configuration** and confirm.

4. If upgrading from Navigator 2.6 (Cloudera Manager 5.7) or lower:

- a. [Start and log into the Cloudera Navigator data management component UI](#). The **Upgrading Navigator** page displays. Depending on the amount of data in the Navigator Metadata Server storage directory, the upgrade process can take *three to four hours* or longer.

When the upgrade is complete, click **Continue**. The Cloudera Navigator landing page is displayed.

5. If you **disabled** any backup or snapshot jobs before the upgrade, now is a good time to re-enable them.
6. The Cloudera Manager upgrade is now **complete**. If Cloudera Manager is not working correctly, or the upgrade did not complete, see [Troubleshooting a Cloudera Manager Upgrade](#) on page 62.

Troubleshooting a Cloudera Manager Upgrade

The Cloudera Manager Server fails to start after upgrade.

The Cloudera Manager Server fails to start after upgrade.

Possible Reasons

There were active commands running before upgrade. This includes commands a user might have run and also commands Cloudera Manager automatically triggers, either in response to a state change, or something configured to run on a schedule, such as Backup and Disaster Recovery replication or snapshot jobs.

Possible Solutions

- Stop any running commands from the Cloudera Manager Admin Console or wait for them to complete. See [Aborting a Pending Command](#).
- Ensure that you have disabled any scheduled **replication or snapshot jobs** from the Cloudera Manager Admin Console to complete before proceeding with the upgrade. See [Enabling, Disabling, or Deleting A Replication Schedule](#).

Re-Running the Cloudera Manager Upgrade Wizard

Minimum Required Role: [Full Administrator](#)

The first time you log in to the Cloudera Manager server after upgrading your Cloudera Manager software, the upgrade wizard runs. If you did not complete the wizard at that time, or if you had hosts that were unavailable at that time and still need to be upgraded, you can re-run the upgrade wizard:

1. Click the **Hosts** tab.
2. Click **Re-run Upgrade Wizard** or **Review Upgrade Status**. This takes you back through the installation wizard to upgrade Cloudera Manager Agents on your hosts as necessary.

3. Select the release of the Cloudera Manager Agent to install. Normally, this is the **Matched Release for this Cloudera Manager Server**. However, if you used a custom repository (instead of `archive.cloudera.com`) for the Cloudera Manager server, select **Custom Repository** and provide the required information. The custom repository allows you to use an alternative location, but that location must contain the matched Agent version.
4. Specify credentials and initiate Agent installation:
 - a. Select **root** for the `root` account, or select **Another user** and enter the username for an account that has password-less `sudo` privileges.
 - b. Select an authentication method:
 - If you choose password authentication, enter and confirm the password.
 - If you choose public-key authentication, provide a passphrase and path to the required key files.

You can modify the default SSH port if necessary.
 - c. Specify the maximum number of host installations to run at once. The default and recommended value is 10. You can adjust this based on your network capacity.
 - d. Click **Continue**.

When you click **Continue**, the Cloudera Manager Agent is upgraded on all the currently managed hosts. You cannot search for new hosts through this process. To add hosts to your cluster, click the **Add New Hosts to Cluster** button.

TLS Protocol Error with OpenJDK

If you are using an older version of OpenJDK 1.8 and have enabled SSL/TLS for the Cloudera Manager Admin Console, you may encounter a TLS protocol error when connecting to the Admin Console, stating that there are no ciphers in common. This is because older versions of OpenJDK may not implement certain TLS ciphers, causing an inability to log into the Cloudera Manager Admin Console when TLS is enabled.

Workaround:

You can workaround this issue by doing one of the following:

- Upgrade OpenJDK to a [supported version of OpenJDK](#) that is higher than version 1.8.0_181.
- If it is not possible to upgrade OpenJDK, enable less secure TLS ciphers in Cloudera Manager. You can do this by opening the `/etc/default/cloudera-scm-server` in a text editor and adding the following line:

```
export CMF_OVERRIDE_TLS_CIPHERS=<cipher_list>
```

Where `<cipher_list>` is a list of TLS cipher suites separated by colons. For example:

```
export
```

Cloudera Bug: OPSAPS-49578

Reverting a Failed Cloudera Manager Upgrade

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

This topic describes how to reinstall the same version of Cloudera Manager you were using previously, so that the version of your Cloudera Manager Agents match the server. The steps below assume that the Cloudera Manager Server is already stopped (because it failed to start after the attempted upgrade).



Important: The following instructions assume that a Cloudera Manager upgrade failed, and that the upgraded server never started, so that the remaining steps of the upgrade process were not performed. The steps below are not sufficient to revert from a running Cloudera Manager deployment.

Ensure Cloudera Manager Server and Agent are stopped.

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Stop the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server stop
```

3. Stop the **Cloudera Manager Agent**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-agent
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent stop
```

Restore the Cloudera Manager Database (if necessary)

If your Cloudera Manager upgrade fails, you need to determine whether the upgrade process has successfully completed updating the schema of the Cloudera Manager database. If the schema update has begun, you must restore the Cloudera Manager database using a [backup](#) taken before you began the upgrade.

1. To determine whether the schema has been updated, examine the Cloudera Manager server logs, and look for a message similar to the following: Updated Schema Version to 60000. (The version number may be different for your environment.)

Run the following command to find the log entry (if the log file is in a different location, substitute the correct path):

```
grep 'Updated Schema Version to ' /var/log/cloudera-scm-server/cloudera-scm-server.log
```

2. If required, restore the database.

The procedure for restoring the database depends on the type of database used by Cloudera Manager.

3. If you are using the embedded PostgreSQL database, stop the Cloudera Manager Embedded PostgreSQL database:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server-db
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server-db stop
```

If you are not using the embedded PostgreSQL database and you attempt to stop it, you might see a message indicating that the service cannot be found. If you see a message that the shutdown failed, then the embedded database is still running, probably because services are connected to the Hive metastore. If the database shutdown fails due to connected services, issue the following command:

RHEL-compatible 7 and higher, Ubuntu 16.04

```
sudo service cloudera-scm-server-db next_stop_fast
sudo service cloudera-scm-server-db stop
```

All other Linux distributions

```
sudo service cloudera-scm-server-db fast_stop
```

Establish Access to the Software

Cloudera Manager needs access to a package repository that contains the updated software packages.

- If the Cloudera Manager hosts have internet access, you can use the publicly available repositories from <https://archive.cloudera.com>. Modify the sample repo files below for your operating system and version.
- If the Cloudera Manager hosts *do not* have internet access, configure [a local package repository](#) hosted on your network. For example: `http://MyWebServer:1234/cloudera-repos`
- Replace `archive.cloudera.com` in the sample repo files below with the URL for your local repository.

Package Repository URL:

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Remove any older files in the existing repository directory:

RHEL / CentOS

```
sudo rm /etc/yum.repos.d/cloudera*manager.repo*
```

SLES

```
sudo rm /etc/zypp/repos.d/cloudera*manager.repo*
```

Debian / Ubuntu

```
sudo rm /etc/apt/sources.list.d/cloudera*.list*
```

3. Fill in the form at the top of this page.
4. Create a repository file so that the package manager can locate and download the binaries:

RHEL / CentOS

Create a file named `/etc/yum.repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
baseurl=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/redhat/7/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

SLES

Create a file named `/etc/zypp/repos.d/cloudera-manager.repo` with the following content:

```
[cloudera-manager]
# Packages for Cloudera Manager
name=Cloudera Manager
```

Upgrading Cloudera Manager

```
baseurl=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/5.15
gpgkey=https://archive.cloudera.com/cm5/sles/12/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

Debian / Ubuntu

Create a file named `/etc/apt/sources.list.d/cloudera_manager.list` with the following content:

```
# Packages for Cloudera Manager
deb https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
deb-src https://archive.cloudera.com/cm5/debian/jessie/amd64/cm/ jessie-cm5.15 contrib
```

```
sudo apt-get update
```

The repository file, as created, refers to the *most recent* maintenance release of the specified minor release. If you would like to use a *specific* maintenance version, for example 5.15.1, replace 5.15 with 5.15.1 in the generated repository file shown above.

5. A Cloudera Manager upgrade can introduce new package dependencies. Your organization may have restrictions or require prior approval for installation of new packages. You can determine which packages may be installed or upgraded:

RHEL / CentOS

```
yum deplist cloudera-manager-agent
```

SLES

```
zypper info --requires cloudera-manager-agent
```

Debian / Ubuntu

```
apt-cache depend cloudera-manager-agent
```

Downgrade the Cloudera Manager Packages



Note: Make sure the repository file above matches the **specific** maintenance version before the upgrade.

1. Downgrade the packages. Note: Only add `cloudera-manager-server-db-2` if you are using the embedded PostgreSQL database.

RHEL / CentOS

```
sudo yum clean all
sudo yum repolist
```

```
sudo yum downgrade "cloudera-manager-*
```

SLES

```
sudo zypper clean --all
```

```
sudo zypper dup -r baseurl
```

Debian / Ubuntu

There is no action that downgrades Cloudera Manager to the version currently in the repository.

2. Verify that you have the correct packages installed.

Debian / Ubuntu

```
dpkg-query -l 'cloudera-manager-*
```

```
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
| / Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name Version Description
+++=-----+-----+-----+
ii cloudera-manager-agent 5.15.0-0.cm...~sq The Cloudera Manager Agent
ii cloudera-manager-daemo 5.15.0-0.cm...~sq Provides daemons for monitoring Hadoop and
related tools.
ii cloudera-manager-serve 5.15.0-0.cm...~sq The Cloudera Manager Server
```

RHEL / CentOS / SLES

```
rpm -qa 'cloudera-manager-*
```

```
cloudera-manager-server-5.15.0-..cm...
cloudera-manager-agent-5.15.0-..cm...
cloudera-manager-daemons-5.15.0-..cm...
cloudera-manager-server-db-2-5.15.0-..cm...
```

Restore the Cloudera Manager Directory

1. Run the following commands to extract the backups:

```
cd $CM_BACKUP_DIR
tar -xf cloudera-scm-agent.tar
tar -xf cloudera-scm-server.tar
```

2. Restore the Cloudera Manager server directory from a backup taken during the upgrade process:

```
sudo -E cp -rp $CM_BACKUP_DIR/etc/cloudera-scm-server/* /etc/cloudera-scm-server
```

```
sudo -E cp -rp $CM_BACKUP_DIR/etc/default/cloudera-scm-server/*
/etc/default/cloudera-scm-server
```

3. If the Cloudera Manager server host has an agent installed, restore the Cloudera Manager agent directory from a backup taken during the upgrade process:

```
sudo -E cp -rp $CM_BACKUP_DIR/etc/cloudera-scm-agent/* /etc/cloudera-scm-agent
```

```
sudo -E cp -rp $CM_BACKUP_DIR/etc/default/cloudera-scm-agent/*
/etc/default/cloudera-scm-agent
```

```
sudo -E cp -rp $CM_BACKUP_DIR/var/run/cloudera-scm-agent/* /var/run/cloudera-scm-agent
```

```
sudo -E cp -rp $CM_BACKUP_DIR/var/lib/cloudera-scm-agent/* /var/lib/cloudera-scm-agent
```

Start Cloudera Manager Again

1. If you are using the embedded PostgreSQL database, start the database:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server-db
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server-db start
```

2. Start the **Cloudera Manager Agent**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-agent
```

If the agent starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent start
```

You should see the following:

```
Starting cloudera-scm-agent: [ OK ]
```

3. Start the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server
```

If the Cloudera Manager Server starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server start
```

You should see the following:

```
Starting cloudera-scm-server: [ OK ]
```

4. Start the **Cloudera Management Service**.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters** > **Cloudera Management Service**.
- c. Select **Actions** > **Start**.



Note: Troubleshooting: If you have problems starting the server, such as database permissions problems, you can use the server's log to troubleshoot the problem.

```
vim /var/log/cloudera-scm-server/cloudera-scm-server.log
```

Upgrading CDH

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

This topic describes how to upgrade CDH from any 5.x version to a higher version of CDH, including major, minor and maintenance releases, using Cloudera Manager and parcels (recommended) or packages.

- **Parcels** – This option uses Cloudera Manager to upgrade CDH and allows you to upgrade your cluster either using a full restart of the cluster, or you can perform a rolling restart if you have HDFS high availability enabled and have a Cloudera Enterprise license. The use of Parcels requires that your cluster be managed by Cloudera Manager. This includes any components that are not a regular part of CDH, such as Spark 2.
- **Packages** – This option is the most time consuming and requires you to log in using ssh and execute a series of package commands on *all hosts* in your cluster. Cloudera recommends that you instead upgrade your cluster using parcels, which allows Cloudera Manager to distribute the upgraded software to all hosts in the cluster without having to log in to each host. If the CDH cluster you are upgrading was installed using packages, you can upgrade it using parcels, and the upgraded version of CDH will then use parcels for future upgrades or changes. You can also [migrate your cluster from using packages to using parcels](#) before starting the upgrade.

There are limitations on the versions of CDH that are eligible for upgrades. See [Supported Upgrade Paths](#) on page 20.

Getting Started Upgrading CDH

Before you upgrade a CDH cluster, you need to gather information, review the limitations and release notes and run some checks on the cluster. See the [Collect Information](#) section below. Fill in the **My Environment** form below to customize your CDH upgrade procedures.

The version of CDH you can upgrade to [depends on the version of Cloudera Manager](#) that is managing the cluster. You may need to [upgrade Cloudera Manager](#) before upgrading CDH.

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)



Warning: Redhat 5 is not a supported operating system for CDH 6.x.



Warning: SLES 11 is not a supported operating system for CDH 6.x.



Warning: Ubuntu 14 and lower are not supported operating systems for CDH 6.x.



Warning: Debian is not a supported operating system for CDH 6.x.



Warning: Ubuntu 12 is not a supported operating systems for CDH 5.16 and higher.



Warning: Kafka in CDH 6.2.0 is based on Apache Kafka 2.1.0, which contains a change to the internal schema used to store consumer offsets. As a result of this change, downgrading Kafka to a version lower than CDH 6.2.0 is **NOT** possible once Kafka has been upgraded to CDH 6.2.0 or higher.



Important: CDH 6.0 supports upgrading from CDH 5.7 or higher, up to 5.14.x, but **NOT** from CDH 5.15.x or higher.



Warning: Backing up data using Cloudera Manager Backup and Disaster Recovery (BDR) does not work when backing up a cluster from Cloudera Manager 5.13 and lower to CDH 6.0 or higher.



Warning: Cloudera Data Science Workbench is not supported with CDH 6.0.x. If you proceed with the upgrade, you will be asked to remove the CDSW service from your cluster. Cloudera Data Science Workbench 1.5 (and higher) is supported with CDH 6.1 (and higher).



Important: CDH 6.x only supports Redhat/Centos 6 or 7.



Important: CDH 6.x only supports SLES 12.



Important: CDH 6.x only supports Ubuntu 16.04 (Xenial).



Important: CDH 6.x does not support Debian.



Important: Oracle 11 is not supported in CDH 6.x. If your deployment uses Oracle 11 as the database for any CDH component, you must upgrade to a [supported database](#) before upgrading CDH.



Warning: If you use a local parcel repository to distribute parcels, be sure to add the type `application/x-gzip` for parcel files in `httpd.conf`: See [Setting Up a Web Server](#) on page 189. Otherwise Cloudera Manager will not recognize any parcel files.

Collect Information

Collect the following information about your environment and fill in the form above. This information will be remembered by your browser on all pages in this Upgrade Guide.

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Run the following command to find the current version of the Operating System:

```
lsb_release -a
```

3. Log in to the Cloudera Manager Admin console and find the following:

- a. The version of Cloudera Manager used in your cluster. Go to **Support > About**.
- b. The version of the JDK deployed in the cluster. Go to **Support > About**.

- c. Whether **High Availability** is enabled for HDFS. Go to the HDFS service and click the **Actions** button. If you see **Disable High Availability**, the cluster has High Availability enabled.
- d. The **Install Method** and Current CDH version. The CDH version number and Install Method are displayed on the Cloudera Manager Home page, to the right of the cluster name.

Preparing to Upgrade CDH

1. You must have SSH access to the Cloudera Manager server hosts and be able to log in using the root account or an account that has password-less sudo permission to all the hosts.
2. Review the [CDH 5 and Cloudera Manager 5 Requirements and Supported Versions](#) or [Cloudera Enterprise 6 Requirements and Supported Versions](#) for the new versions you are upgrading to. If your hosts require an operating system upgrade, you must perform the upgrade before upgrading CDH. See [Upgrading the Operating System](#) on page 29.
3. Ensure that Java 1.7 or 1.8 is installed across the cluster. Cloudera Manager 6.0.0 and higher and CDH 6.0.0 and higher require Java 1.8. For installation instructions and recommendations, see [Upgrading the JDK](#) on page 22.
4. Review the following documents:

CDH 5

- [CDH 5 Release Notes](#):
 - [Incompatible Changes](#).
 - [Known Issues](#).
 - [Issues Fixed](#).
- [Cloudera Security Bulletins](#).

CDH 6

- [CDH 6 Release Notes](#):
 - [Install and Upgrade Notes](#) on page 14
 - [New Features in CDH 6.0.0](#)
 - [Fixed Issues in CDH 6.0.0](#)
 - [Known Issues and Limitations in CDH 6.0.0](#)
 - [Unsupported Features in CDH 6.0.0](#)
 - [Incompatible Changes in CDH 6.0.0](#)
 - [Cloudera Security Bulletins](#)
5. Review the upgrade procedure and reserve a maintenance window with enough time allotted to perform all steps. For production clusters, Cloudera recommends allocating up to a full day maintenance window to perform the upgrade, depending on the number of hosts, the amount of experience you have with Hadoop and Linux, and the particular hardware you are using.
 6. If you are upgrading from CDH 5.1 or lower, and use Hive Date partition columns, you might need to update the date format. See [Date Partition Columns](#).
 7. If the cluster uses **Impala**, check your SQL against the newest reserved words listed in [incompatible changes](#). If upgrading across multiple versions, or in case of any problems, check against the full list of [Impala keywords](#).
 8. Run the [Security Inspector](#) and fix any reported errors.

Go to **Administration > Security > Security Inspector**.

9. Log in to any cluster node as the `hdfs` user, run the following commands, and correct any reported errors:

```
hdfs fsck / -includeSnapshots
```



Note: The `fsck` command might take 10 minutes or more to complete, depending on the number of files in your cluster.

```
hdfs dfsadmin -report
```

See [HDFS Commands Guide](#) in the Apache Hadoop documentation.

- 10 Log in to any DataNode as the `hdfs` user, run the following command, and correct any reported errors:

```
hbase hbck
```

See [Checking Consistency in HBase Tables](#).

- 11 If your cluster uses HBase, see [Migrating Apache HBase Before Upgrading to CDH 6](#) on page 87.
- 12 If the cluster uses Kudu, log in to any cluster host and run the `ksck` command as the `kudu` user (`sudo -u kudu`). If the cluster is Kerberized, first `kinit` as `kudu` then run the command:

```
kudu cluster ksck <master_addresses>
```

For the full syntax of this command, see [Checking Cluster Health with `ksck`](#).

- 13 If you have configured **Hue** to use TLS/SSL and you are upgrading from CDH 5.2 or lower to CDH 5.3 or higher, Hue validates CA certificates and requires a truststore. To create a truststore, follow the instructions in [Hue as a TLS/SSL Client](#).
- 14 If you are upgrading to CDH 6.0 or higher, and Hue is deployed in the cluster, and Hue is using PostgreSQL as its database, you must manually install `psycpg2`. See [Installing Dependencies for Hue](#) on page 89.
- 15 If your cluster uses the **Flume Kafka client**, and you are upgrading to CDH 5.8.0 or CDH 5.8.1, perform the extra steps described in [Upgrading to CDH 5.8.0 or CDH 5.8.1 When Using the Flume Kafka Client](#) on page 131 and then continue with the procedures in this topic.
- 16 If your cluster uses **Impala and Llama**, this role has been deprecated as of CDH 5.9 and you must remove the role from the Impala service before starting the upgrade. If you do not remove this role, the upgrade wizard will halt the upgrade.

To determine if Impala uses Llama:

1. Go to the Impala service.
2. Select the **Instances** tab.
3. Examine the list of roles in the **Role Type** column. If Llama appears, the Impala service is using Llama.

To remove the Llama role:

1. Go to the Impala service and select **Actions > Disable YARN and Impala Integrated Resource Management**.

The **Disable YARN and Impala Integrated Resource Management** wizard displays.

2. Click **Continue**.

The **Disable YARN and Impala Integrated Resource Management Command** page displays the progress of the commands to disable the role.

3. When the commands have completed, click **Finish**.

- 17 If your cluster uses **Sentry**, and are upgrading from CDH 5.12 or lower, you might need to increase the Java heap memory for Sentry. See [Performance Guidelines](#).
- 18 If your cluster uses **Sentry** and an Oracle database, and you are upgrading from CDH 5.13.0 or higher to CDH 5.16.0 or higher, you must manually add the `AUTHZ_PATH.AUTHZ_OBJ_ID` index if it does not already exist. Adding the index manually decreases the time Sentry takes to get a full snapshot for HDFS sync. Use the following command to add the index:

```
CREATE INDEX "AUTHZ_PATH_FK_IDX" ON "AUTHZ_PATH" ("AUTHZ_OBJ_ID");
```


19 The following services are no longer supported as of Enterprise 6.0.0:

- Accumulo
- Sqoop 2
- MapReduce 1
- Spark 1.6
- Record Service

You must stop and delete these services before upgrading CDH. See [Stopping a Service on All Hosts](#) and [Deleting Services](#).

20 Open the Cloudera Manager Admin console and collect the following information about your environment:

- a. The version of Cloudera Manager. Go to **Support > About**.
- b. The version of the JDK deployed. Go to **Support > About**.
- c. The version of CDH and whether the cluster was installed using parcels or packages. It is displayed next to the cluster name on the **Home** page.
- d. The services enabled in your cluster.

Go to **Clusters > Cluster name**.

- e. Whether HDFS High Availability is enabled.

Go to **Clusters** click **HDFS Service**, click **Actions** menu. It is enabled if you see an menu item **Disable High Availability**.

21 Back up Cloudera Manager before beginning the upgrade. See [Backing Up Cloudera Manager](#) on page 41.

22 Review all [CDH 6 pre-upgrade migration steps](#). There are steps you must perform before beginning the upgrade for the following components: Sentry, Cloudera Search, Apache Spark, HBase, Hue, Key Trustee KMS, HSM KMS.

Backing Up CDH

This topic describes how to back up a CDH cluster managed by Cloudera Manager prior to upgrading the cluster. These procedures *do not back up the data stored in the cluster*. Cloudera recommends that you maintain regular backups of your data using the Backup and Disaster Recovery features of Cloudera Manager. See [Backup and Disaster Recovery](#).

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

Backing up CDH before you upgrade allows you to roll back the upgrade if necessary. See [Rolling Back a CDH 5 to CDH 6 Upgrade](#) on page 132

The following CDH components do not require backups:

- MapReduce
- YARN
- Spark
- Pig
- Impala

Complete the following backup steps before upgrading CDH:

Back Up HDFS Metadata on the NameNode

[Not required for CDH maintenance release upgrades.]

The steps in this section are only required for the following upgrades:

- CDH 5.0 or 5.1 to 5.2 or higher
- CDH 5.2 or 5.3 to 5.4 or higher

Back up HDFS metadata using the following command:

```
hdfs dfsadmin -fetchImage myImageName
```

Back Up the Repository Files

Back up the repository files on all cluster hosts.

The `tar` commands in the steps below may return the following message. It is safe to ignore this message:

```
tar: Removing leading `/' from member names
```

RHEL / CentOS

```
sudo -E tar -cf CDH_BACKUP_DIR/repository-CM-CDH.tar /etc/yum.repos.d
```

SLES

```
sudo -E tar -cf CDH_BACKUP_DIR/repository-CM-CDH.tar /etc/zypp/repos.d
```

Debian / Ubuntu

```
sudo -E tar -cf CDH_BACKUP_DIR/repository-CM-CDH.tar /etc/apt/sources.list.d
```

Back Up Databases



Warning: Backing up databases requires that you *stop some services*, which might make them unavailable during backup.

Gather the following information:

- Type of database (PostgreSQL, Embedded PostgreSQL, MySQL, MariaDB, or Oracle)
- Hostnames of the databases
- Database names
- Port number used by the databases
- Credentials for the databases

Open the Cloudera Manager Admin Console to find the database information for any of the following services you have deployed in your cluster:

- **Sqoop, Oozie, and Hue** – Go to **Cluster Name** > **Configuration** > **Database Settings**.



Note: The Sqoop Metastore uses a HyperSQL (HSQLDB) database. See the [HyperSQL](#) documentation for backup procedures.

- **Hive Metastore** – Go to the Hive service, select **Configuration**, and select the **Hive Metastore Database** category.
- **Sentry** – Go to the Sentry service, select **Configuration**, and select the **Sentry Server Database** category.

To back up the databases

Perform the following steps for each database you back up:

1. If not already stopped, stop the service. If Cloudera Manager indicates that there are dependent services, also stop the dependent services.

- a. On the **Home > Status** tab, click



to the right of the service name and select **Stop**.

- b. Click **Stop** in the next screen to confirm. When you see a **Finished** status, the service has stopped.

2. Back up the database. Substitute the database name, hostname, port, user name, and backup directory path and run the following command:

MySQL

```
mysqldump --databases database_name --host=database_hostname --port=database_port -u database_username -p > backup_directory_path/database_name-backup-`date +%F`-CDH.sql
```

PostgreSQL/Embedded

```
pg_dump -h database_hostname -U database_username -W -p database_port database_name > backup_directory_path/database_name-backup-`date +%F`-CDH.sql
```

Oracle

Work with your database administrator to ensure databases are properly backed up.

For additional information about backing up databases, see these vendor-specific links:

- MariaDB 5.5: <http://mariadb.com/kb/en/mariadb/backup-and-restore-overview/>
- MySQL 5.5: <http://dev.mysql.com/doc/refman/5.5/en/backup-and-recovery.html>
- MySQL 5.6: <http://dev.mysql.com/doc/refman/5.6/en/backup-and-recovery.html>
- PostgreSQL 8.4: <https://www.postgresql.org/docs/8.4/static/backup.html>
- PostgreSQL 9.2: <https://www.postgresql.org/docs/9.2/static/backup.html>
- PostgreSQL 9.3: <https://www.postgresql.org/docs/9.3/static/backup.html>
- Oracle 11gR2: http://docs.oracle.com/cd/E11882_01/backup.112/e10642/toc.htm
- HyperSQL: http://hsqldb.org/doc/guide/management-chapt.html#mtc_backup

3. Start the service.

- a. On the **Home > Status** tab, click



to the right of the service name and select **Start**.

- b. Click **Start** in the next screen to confirm. When you see a **Finished** status, the service has started.

Back Up ZooKeeper

On all ZooKeeper hosts, back up the ZooKeeper data directory specified with the `dataDir` property in the ZooKeeper configuration. The default location is `/var/lib/zookeeper`. For example:

```
cp -rp /var/lib/zookeeper/ /var/lib/zookeeper-backup-`date +%F`CM-CDH
```

To identify the ZooKeeper hosts, open the Cloudera Manager Admin console and go to the ZooKeeper service and click the **Instances** tab.

Record the permissions of the files and directories; you will need these to roll back ZooKeeper.

Back Up HDFS

Follow this procedure to back up an HDFS deployment.



Note: To locate the hostnames required to backup HDFS (for JournalNodes, DataNodes, and NameNodes), open the Cloudera Manager Admin Console, go to the HDFS service, and click the **Instances** tab.

1. If high availability is enabled for HDFS, run the following command on all hosts running the JournalNode role:

```
cp -rp /dfs/jn /dfs/jn-CM-CDH
```

2. On all NameNode hosts, back up the NameNode runtime directory. Run the following commands:

```
mkdir -p /etc/hadoop/conf.rollback.namenode
```

```
cd /var/run/cloudera-scm-agent/process/ && cd `ls -tl | grep -e "-NAMENODE\$" | head -1`
```

```
cp -rp * /etc/hadoop/conf.rollback.namenode/
```

```
rm -rf /etc/hadoop/conf.rollback.namenode/log4j.properties
```

```
cp -rp /etc/hadoop/conf.cloudera.HDFS_service_name/log4j.properties  
/etc/hadoop/conf.rollback.namenode/
```

These commands create a temporary rollback directory. If a rollback to CDH 5.x is required later, the rollback procedure requires you to modify files in this directory.

3. Back up the runtime directory for all DataNodes. Run the following commands on all DataNodes:

```
mkdir -p /etc/hadoop/conf.rollback.datanode/
```

```
cd /var/run/cloudera-scm-agent/process/ && cd `ls -tl | grep -e "-DATANODE\$" | head -1`
```

```
cp -rp * /etc/hadoop/conf.rollback.datanode/
```

```
rm -rf /etc/hadoop/conf.rollback.datanode/log4j.properties
```

```
cp -rp /etc/hadoop/conf.cloudera.HDFS_service_name/log4j.properties  
/etc/hadoop/conf.rollback.datanode/
```

4. If high availability is *not* enabled for HDFS, backup the runtime directory of the Secondary NameNode. Run the following commands on all Secondary NameNode hosts:

```
mkdir -p /etc/hadoop/conf.rollback.secondarynamenode/
```

```
cd /var/run/cloudera-scm-agent/process/ && cd `ls -tl | grep -e "-SECONDARYNAMENODE\$" | head -1`
```

```
cp -rp * /etc/hadoop/conf.rollback.secondarynamenode/
```

```
rm -rf /etc/hadoop/conf.rollback.secondarynamenode/log4j.properties
```

```
cp -rp /etc/hadoop/conf.cloudera.HDFS_service_name /log4j.properties  
/etc/hadoop/conf.rollback.secondarynamenode/
```

Back Up Key Trustee Server and Clients

For the detailed procedure, see [Backing Up and Restoring Key Trustee Server and Clients](#).

Back Up HSM KMS

When running the HSM KMS in high availability mode, if either of the two nodes fails, a role instance can be assigned to another node and federated into the service by the single remaining active node. In other words, you can bring a node that is part of the cluster, but that is not running HSM KMS role instances, into the service by making it an HSM KMS role instance—more specifically, an HSM KMS proxy role instance and an HSM KMS metastore role instance. So each node acts as an online ("hot" backup) backup of the other. In many cases, this will be sufficient. However, if a manual ("cold" backup) backup of the files necessary to restore the service from scratch is desirable, you can create that as well.

To create a backup, copy the `/var/lib/hsmkp` and `/var/lib/hsmkp-meta` directories on one or more of the nodes running HSM KMS role instances.

To restore from a backup: bring up a completely new instance of the HSM KMS service, and copy the `/var/lib/hsmkp` and `/var/lib/hsmkp-meta` directories from the backup onto the file system of the restored nodes before starting HSM KMS for the first time.

Back Up Navigator Encrypt

It is recommended that you back up Navigator Encrypt configuration directory after installation, and again after any configuration updates.

1. To manually back up the Navigator Encrypt configuration directory (`/etc/navencrypt`):

```
$ zip -r --encrypt nav-encrypt-conf.zip /etc/navencrypt
```

The `--encrypt` option prompts you to create a password used to encrypt the zip file. This password is also required to decrypt the file. Ensure that you protect the password by storing it in a secure location.

2. Move the backup file (`nav-encrypt-conf.zip`) to a secure location.



Warning: Failure to back up the configuration directory makes your backed-up encrypted data unrecoverable in the event of data loss.

Back Up HBase

Because the rollback procedure also rolls back HDFS, the data in HBase is also rolled back. In addition, HBase metadata stored in ZooKeeper is recovered as part of the ZooKeeper rollback procedure.

If your cluster is configured to use HBase replication, Cloudera recommends that you document all replication peers. If necessary (for example, because the HBase znode has been deleted), you can roll back HBase as part of the HDFS rollback without the ZooKeeper metadata. This metadata can be reconstructed in a fresh ZooKeeper installation, with the exception of the replication peers, which you must add back. For information on enabling HBase replication, listing peers, and adding a peer, see [HBase Replication](#) in the CDH 5 documentation.

Back Up Search

Back up your Solr metadata using the following procedure. This procedure allows you to roll back to the pre-upgrade state if any problems occur during the upgrade process.

1. Make sure that the HDFS and ZooKeeper services are running.
2. Stop the Solr service (**Solr service > Actions > Stop**). If you see a message about stopping dependent services, click **Cancel** and stop the dependent services first, and then stop the Solr service.
3. Back up the Solr configuration metadata (**Solr service > Actions > Backup Solr Configuration Meta-data for Upgrade**). Make sure that the directory you specified for the **Upgrade Backup Directory** configuration property exists in HDFS and is writable by the Search superuser (`solr` by default).

4. Start the Solr service (**Solr service > Actions > Start**).
5. Start any dependent services that you stopped.

Back Up Sqoop 2



Note: Sqoop 2 is not supported in CDH 6.x. To upgrade to CDH 6.x, you must delete the Sqoop 2 service before the upgrade. Perform these steps before deleting the Sqoop 2 service.

If you are not using the default embedded Derby database for Sqoop 2, back up the database you have configured for Sqoop 2. Otherwise, back up the `repository` subdirectory of the Sqoop 2 metastore directory. This location is specified with the **Sqoop 2 Server Metastore Directory** property. The default location is: `/var/lib/sqoop2`. For this default location, Derby database files are located in `/var/lib/sqoop2/repository`.

Back Up Hue

1. On all hosts running the Hue Server role, back up the app registry file:

Parcel installations

```
mkdir -p /opt/cloudera/parcels_backup
cp -rp /opt/cloudera/parcels/CDH/lib/hue/app.reg
/opt/cloudera/parcels_backup/app.reg-CM-CDH
```

Package installations

```
cp -rp /usr/lib/hue/app.reg /usr/lib/hue_backup/app.reg-CM-CDH
```

CDH 6 Pre-Upgrade Migration Steps

If you have deployed the Sentry, HBase, Cloudera Search, Spark, Key Trustee KMS, or HSM KMS services in a CDH 5.x cluster that you want to upgrade to CDH 6, see the following topics for additional pre-upgrade steps:

Migrating from Sentry Policy Files to the Sentry Service

If your cluster uses Sentry policy file authorization, you must migrate the policy files to the database-backed Sentry service before you upgrade to CDH 6.

Complete the following steps to upgrade from Sentry policy files to the database-backed Sentry service:

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

1. Disable the existing Sentry policy file for any Hive, Impala, or Solr services on the cluster. To do this:
 - a. Go to the Hive, Impala, or Solr service.
 - b. Click the **Configuration** tab.
 - c. Select **Scope > Service Name (Service-Wide)**.
 - d. Select **Category > Policy File Based Sentry**.
 - e. Clear **Enable Sentry Authorization using Policy Files**. Cloudera Manager throws a validation error if you attempt to configure the Sentry service while this property is checked.
 - f. Repeat for any remaining Hive, Impala, or Solr services.
2. Add the new Sentry service to your cluster. For instructions, see [Adding the Sentry Service](#).
3. To begin using the Sentry service, see [Enabling the Sentry Service Using Cloudera Manager](#) and [Configuring the Sentry Service](#).
4. (Optional) Use command line tools to migrate existing policy file grants.

- If you want to migrate existing Sentry configurations for Solr, use the `solrctl sentry --convert-policy-file` command, described in [solrctl Reference](#).
 - For Hive and Impala, use the command-line interface Beeline to issue grants to the Sentry service to match the contents of your old policy file(s). For more details on the Sentry service and examples on using Grant/Revoke statements to match your policy file, see [Hive SQL Syntax for Use with Sentry](#).
5. Restart the affected services as described in [Restarting Services and Instances after Configuration Changes](#) to apply the changes.

Migrating Cloudera Search Configuration Before Upgrading to CDH 6

Because Cloudera Search is included in CDH, upgrading CDH upgrades Cloudera Search. If you are upgrading to CDH 6 from CDH 5, and you are using Cloudera Search, you must complete some preparatory work.

Cloudera Search in CDH 6 uses Apache Solr 7, which has some incompatibilities with previous Solr versions. To facilitate the upgrade, Cloudera provides a Solr configuration migration script, `solr-upgrade.sh`. This script is included with Cloudera Manager 6 and CDH 6. You must [upgrade to Cloudera Manager 6](#) before you can complete these procedures.

The following topics describe the steps and procedures to upgrade to CDH 6 if you are using Cloudera Search:

Before You Begin



Warning: Due to changes in the underlying Apache Lucene file format between the CDH 5 and CDH 6 versions of Solr, it is not possible to directly upgrade your existing indexes. The following procedures update only your Solr configuration and metadata to be compatible with the new Solr version. After upgrading to CDH 6, you must re-index your collections.

Make sure that the source data that was used to index your collections is still available before upgrading. Plan downtime for any applications or services that use your Search deployment.

Although the configuration migration script addresses known incompatibilities, there might be incompatible configurations that are not detected by the script. Do not upgrade your production Cloudera Search environments to CDH 6 until you have tested the migrated configuration with CDH 6 on a testing or development cluster.

Before upgrading:

- Make sure you are running Cloudera Manager 6.0 or higher. For instructions on upgrading to Cloudera Manager 6, see [Upgrading Cloudera Manager](#) on page 38.
- If you are using Apache Sentry with policy files, complete the steps in [Migrating from Sentry Policy Files to the Sentry Service](#) on page 78.
- Stop making changes to your Cloudera Search environment. Make sure that no configuration changes are made to Cloudera Search for the duration of the migration and upgrade. This includes adding or removing Solr Server hosts, moving Solr Server roles between hosts, changing hostnames, and so on.
- Plan for an outage for any applications or services that use your Search deployment until you have completed re-indexing after the upgrade.
- If you are using Kerberos, create a `jaas.conf` file for the Search service superuser (`solr` by default). For instructions on creating a `jaas.conf` file, see [Configuring a jaas.conf File](#).
- If you are using the Lily HBase Indexer service, stop writing to any tables that are indexed into Solr, and stop the Lily HBase Indexer service (**Key-Value Store Indexer service** > **Actions** > **Stop**).
- If you are using the Lily HBase Indexer service with Apache Sentry policy files:
 - If you are upgrading from a CDH version lower than 5.14.0, disable Sentry for the Lily HBase Indexer service before upgrading. To do so, uncheck the box labeled **Enable Sentry Authorization using Policy Files (Key-Value Store Indexer service > Configuration > Category > Policy File Based Sentry)**.
 - If you are upgrading from CDH 5.14.0 or higher, migrate to the Sentry Service before upgrade. CDH versions lower than 5.14.0 do not support using the Sentry Service with the HBase Indexer service. For instructions, see [Migrating HBase Indexer Sentry Policy Files to the Sentry Service](#).

- Do not create, delete, or modify any collections for the duration of the migration and upgrade.

You can continue indexing to existing collections (except Lily HBase indexing) until otherwise instructed.

Solr Configuration Migration Script Overview



Warning: Although the configuration migration script addresses known incompatibilities, there might be incompatible configurations that are not detected by the script. Do not upgrade your production Cloudera Search environments to CDH 6 until you have tested the migrated configuration with CDH 6 on a testing or development cluster.

Despite widespread enterprise adoption, Solr lacks automated upgrade tooling. It has long been a challenge to understand the implications of a Solr upgrade. Solr admins were required to review the Solr release notes and manually identify configuration changes needed to address incompatibilities or to take advantage of new features. Additionally, admins had to determine whether they could upgrade existing indexes, or if they had to re-index the raw data.

Starting in Cloudera Enterprise 6, Cloudera provides a Solr configuration migration script to simplify the upgrade process by providing upgrade instructions tailored to your configuration. These instructions can help you to answer following questions:

- Does my Solr configuration use any configurations that are incompatible with the new version? If so, which ones?
- For each incompatibility, what do I need to do to address it? Where can I get more information about this incompatibility, and why it was introduced?
- Are there any changes in Lucene or Solr that require me to do a full re-index, or is it sufficient to upgrade the index? For all upgrades to CDH 6 from CDH 5, re-indexing is required.

This tool is built using the [Extensible Stylesheet Language Transformations](#) engine. The upgrade rules, implemented as XSLT transformations, can identify incompatibilities and in some cases can fix them automatically.

In general, incompatibilities are categorized as follows:

- **ERROR:** The removal of a Lucene or Solr configuration element (such as a field type) is marked as **ERROR** in the validation output. These types of incompatibilities typically result in failure to start the Solr service or load the core. To address this, you must manually fix the Solr configuration.
- **WARNING:** The deprecation of a configuration element in the new Solr version is marked as **WARNING** in the validation output. In general, these types of incompatibilities do not prevent starting the Solr service or loading cores, but may prevent applications from using new Lucene or Solr features (or bug fixes). You can choose to make changes to Solr configuration using application specific knowledge to fix such incompatibility.
- **INFO:** Incompatibilities that can be fixed automatically by rewriting the Solr configuration are marked **INFO** in the validation output. This can include incompatibilities in the underlying Lucene implementation (for example, [LUCENE-6058](#)) that would require rebuilding the index instead of upgrading it. Typically, these incompatibilities do not result in failure to start the Solr service or load cores, but may affect query result accuracy or consistency of the underlying indexed data.

Running the Solr Configuration Migration Script



Warning: Do not run the configuration migration script (`solr-upgrade.sh`) or implement its recommended changes until you are ready to upgrade to CDH 6. Always test upgrades in a development or testing environment before upgrading your production environments. This is especially important when upgrading to CDH 6 due to the incompatibilities between Solr 4 (used in CDH 5) and Solr 7 (used in CDH 6).

Running the script against a Solr service in CDH 5 and implementing its recommendations will modify your configuration to be compatible with Solr 7, which is incompatible with the Solr versions used in CDH 5. After completing these procedures, your Solr service will be in an unstable state until you complete the upgrade to CDH 6.

The Solr configuration migration script, `solr-upgrade.sh`, is included with CDH 6 and Cloudera Manager 6 agent software. This enables you to run the script after upgrading to Cloudera Manager 6, but before upgrading to CDH 6.

Make sure to run the script on a host that is assigned a **Solr Server** or Solr service **Gateway** role. Confirm that the `SOLR_ZK_ENSEMBLE` environment variable is set in `/etc/solr/conf/solr-env.sh`:

```
cat /etc/solr/conf/solr-env.sh
```

```
export
SOLR_ZK_ENSEMBLE=zk01.example.com:2181,zk02.example.com:2181,zk03.example.com:2181/solr
export SENTRY_CONF_DIR=/etc/solr/conf.cloudera.SOLR-1/sentry-conf
```

The script is located at:

- **Cloudera Manager 6 Agent:** `/opt/cloudera/cm/solr-upgrade/solr-upgrade.sh`
- **CDH 6 Parcels:** `/opt/cloudera/parcels/CDH/lib/solr/solr-upgrade/solr-upgrade.sh`
- **CDH 6 Packages:** `/usr/lib/solr/solr-upgrade/solr-upgrade.sh`

When running the script included with Cloudera Manager 6 Agent, you must specify the location of the CDH 5 Solr binaries using the `CDH_SOLR_HOME` environment variable. If you are using parcels, Solr binaries are located at `/opt/cloudera/parcels/CDH/lib/solr`. For package installations, the location is `/usr/lib/solr`.

For example:

```
export CDH_SOLR_HOME=/opt/cloudera/parcels/CDH/lib/solr
```

For your reference, the `solr-upgrade.sh` command syntax is as follows. The appropriate command arguments are provided in later steps.

```
./solr-upgrade.sh help
```

```
Usage: ./solr-upgrade.sh command [command-arg]Options:
```

```
--zk zk_ensemble
--jaas jaas.conf
--debug Prints error output of calls
--trace Prints executed commands
```

```
Commands:
```

```
help
download-metadata -d dest_dir
validate-metadata -c metadata_dir
bootstrap-config -c metadata_dir
config-upgrade [--dry-run] -c conf_path -t conf_type -u upgrade_processor_conf -d
result_dir [-v]
bootstrap-collections -c metadata_folder_path -d local_work_dir -h hdfs_work_dir
```

```
Parameters:
```

```
-c <arg>      This parameter specifies the path of Solr configuration to be operated
upon.
-t <arg>      This parameter specifies the type of Solr configuration to be validated
and
solr.xml      transformed.The tool currently supports schema.xml, solrconfig.xml and
-d <arg>      This parameter specifies the directory path where the result of the
command      should be stored.
-h <arg>      This parameter specifies the HDFS directory path where the result of the
command      should be stored on HDFS. Eg. /solr-backup
-u <arg>      This parameter specifies the path of the Solr upgrade processor
configuration.
--dry-run     This command will perform compatibility checks for the specified Solr
configuration.
-v           This parameter enables printing XSLT compiler warnings on the command
output.
```

Migrating Cloudera Search Configuration for Compatibility with CDH 6



Warning: Do not run the configuration migration script (`solr-upgrade.sh`) or implement its recommended changes until you are ready to upgrade to CDH 6. Always test upgrades in a development or testing environment before upgrading your production environments. This is especially important when upgrading to CDH 6 due to the incompatibilities between Solr 4 (used in CDH 5) and Solr 7 (used in CDH 6).

Running the script against a Solr service in CDH 5 and implementing its recommendations will modify your configuration to be compatible with Solr 7, which is incompatible with the Solr versions used in CDH 5. After completing these procedures, your Solr service will be in an unstable state until you complete the upgrade to CDH 6.

Use the following procedures to migrate your Cloudera Search configuration and upgrade to CDH 6. The provided migration script cannot upgrade the Lucene index files. After upgrading, you must re-index your collections. For more information, see [Reindexing in Solr](#) in the Apache Solr wiki. The upgrade process is as follows:

Set Configuration Properties in Cloudera Manager



Note: If you have multiple Solr services in the same CDH cluster, you must configure these properties for each Solr service.

As part of the CDH 6 upgrade, Cloudera Manager backs up and validates your migrated Solr configuration to ensure that it is compatible with CDH 6. For the upgrade to succeed, you must designate one of your Solr Server hosts to perform these actions, and specify HDFS and local directories for the backup and migrated configuration files, respectively:

1. In the Cloudera Manager Admin Console, go to **Solr service > Configuration**.
2. In the **Search** field, type `upgrade` to filter the configuration parameters.
3. For the **Solr Server for Upgrade** property, select one of your **Solr Server** hosts. When you run the migration script, make sure to run it on this host.
4. For the **Upgrade Backup Directory** property, specify a directory in HDFS. This directory must exist and be readable by the Solr service superuser (`solr` by default). Examples in these procedures use `/cdh6-solr-upgrade/backup` for this HDFS directory.
5. For the **Upgrade Metadata Directory** property, specify a directory on the local filesystem of the host you selected as the **Solr Server for Upgrade**. When you run the migration script, make sure that you copy the migrated configuration to this directory. This directory must also be readable by the Solr service superuser. Examples in these procedures use `/cdh6-solr-metadata/migrated-config` for this local directory.
6. Enter a **Reason for change**, and then click **Save Changes** to commit the changes.

Back Up Solr Configuration and Data

Before upgrading to CDH 6, back up your Solr collections using the following procedure. This allows you to roll back to the pre-upgrade state if any problems occur during the upgrade process.

1. Make sure that the HDFS and ZooKeeper services are running.
2. Stop the Solr service (**Solr service > Actions > Stop**). If you see a message about stopping dependent services, click **Cancel** and stop the dependent services first, and then stop the Solr service.
3. Back up the Solr configuration metadata (**Solr service > Actions > Backup Solr Configuration Meta-data for Upgrade**). Make sure that the directory you specified for the **Upgrade Backup Directory** configuration property exists in HDFS and is writable by the Search superuser (`solr` by default).
4. Start the Solr service (**Solr service > Actions > Start**).
5. Start any dependent services that you stopped.

Migrate the Configuration

The migration tool supports migrating `schema.xml` (and `managed-schema`), `solrconfig.xml`, and `solr.xml` configuration files. Run these commands on the host you selected as the **Solr Server for Upgrade** in [Set Configuration Properties in Cloudera Manager](#) on page 82.

1. Set the `CDH_SOLR_HOME` environment variable for your installation method:

- **Parcels:**

```
export CDH_SOLR_HOME=/opt/cloudera/parcels/CDH/lib/solr
```

- **Packages:**

```
export CDH_SOLR_HOME=/usr/lib/solr
```

2. Set the `JAVA_HOME` environment variable to the JDK you are using. For example:

```
export JAVA_HOME="/usr/java/jdk1.8.0_141-cloudera"
```

3. Create a working directory for the migration:

```
mkdir $HOME/cdh6-solr-migration
```

4. If you have enabled [Kerberos](#), run `kinit` with the Solr service superuser. For example:

```
kinit solr@EXAMPLE.COM
```

5. Download the current (CDH 5) Solr configuration from ZooKeeper:

```
/opt/cloudera/cm/solr-upgrade/solr-upgrade.sh download-metadata -d  
$HOME/cdh6-solr-migration
```

If you have enabled Kerberos and configured [ZooKeeper access control lists](#) (ACLs), specify your [JAAS configuration file](#) by adding the `--jaas` parameter to the command. For example:

```
/opt/cloudera/cm/solr-upgrade/solr-upgrade.sh --jaas $HOME/solr-jaas.conf  
download-metadata -d $HOME/cdh6-solr-migration
```

6. Initialize a directory for the migrated configuration as a copy of the current config:

```
cp -r $HOME/cdh6-solr-migration $HOME/cdh6-migrated-solr-config
```

7. Migrate the `solr.xml` file:

- a. Run the migration script:

```
/opt/cloudera/cm/solr-upgrade/solr-upgrade.sh config-upgrade -t solrxml -c  
$HOME/cdh6-solr-migration/solr.xml -u  
/opt/cloudera/cm/solr-upgrade/validators/solr_4_to_7_processors.xml -d /tmp
```

If you have enabled Kerberos, specify your JAAS configuration file by appending `--jaas` `/path/to/solr-jaas.conf` to the command.

- b. If the script reports any incompatibilities, fix them in the working directory (`$HOME/cdh6-solr-migration/solr.xml` in this example) and then re-run the script. Each time you run

the script, the files in the output directory (/tmp in this example) are overwritten. Repeat until the script outputs no incompatibilities and the solr.xml migration is successful. For example:

```
Validating solrxml...
No configuration errors found...
No configuration warnings found...

Following incompatibilities will be fixed by auto-transformations (using --upgrade
command):
  * System property used to define SOLR server port has changed from solr.port to
jetty.port

Solr solrxml validation is successful. Please review /tmp/solrxml_validation.html for
more details.

Applying auto transformations...

The upgraded configuration file is available at /tmp/solr.xml
```

8. Copy the migrated solr.xml file to the migrated configuration directory:

```
cp /tmp/solr.xml $HOME/cdh6-migrated-solr-config
```

9. For each collection configuration set in \$HOME/cdh6-solr-migration/configs/, migrate the configuration and schema. Each time you run the script, the output file is overwritten. Do not proceed to the next collection until the migration is successful and you have copied the migrated file to its final destination.

a. Run the migration script for solrconfig.xml. For example, for a configuration set named tweets_config:

```
/opt/cloudera/cm/solr-upgrade/solr-upgrade.sh config-upgrade -t solrconfig -c
$HOME/cdh6-solr-migration/configs/tweets_config/conf/solrconfig.xml -u
/opt/cloudera/cm/solr-upgrade/validators/solr_4_to_7_processors.xml -d /tmp
```

b. If the script reports any incompatibilities, fix them in the working directory (\$HOME/cdh6-solr-migration/configs/tweets_config/conf/solrconfig.xml in this example) and then re-run the script. Repeat until the script outputs no incompatibilities and the solrconfig.xml migration is successful. You should see a message similar to the following:

```
Solr solrconfig validation is successful. Please review /tmp/solrconfig_validation.html
for more details.

Applying auto transformations...

The upgraded configuration file is available at /tmp/solrconfig.xml
```

c. Copy the migrated solrconfig.xml file to the collection configuration directory in the migrated directory. For example:

```
cp /tmp/solrconfig.xml $HOME/cdh6-migrated-solr-config/configs/tweets_config/conf/
```

d. Run the migration script for schema.xml (or managed-schema). For example, for a configuration set named tweets_config:

```
/opt/cloudera/cm/solr-upgrade/solr-upgrade.sh config-upgrade -t schema -c
$HOME/cdh6-solr-migration/configs/tweets_config/conf/schema.xml -u
/opt/cloudera/cm/solr-upgrade/validators/solr_4_to_7_processors.xml -d /tmp
```

e. If the script reports any incompatibilities, fix them in the working directory (\$HOME/cdh6-solr-migration/configs/tweets_config/conf/schema.xml in this example) and then re-run the script. Repeat until the script outputs no incompatibilities and the solrconfig.xml migrations are all successful.

- f. Copy the migrated `schema.xml` file to the collection configuration directory in the migrated directory. For example:

```
cp /tmp/schema.xml $HOME/cdh6-migrated-solr-config/configs/tweets_config/conf/
```

- g. Repeat for all configuration sets in `$HOME/cdh6-solr-migration/configs/`.

Validate the Migrated Configuration

The `solr-upgrade.sh` script includes a `validate-metadata` command that you can run against the migrated Solr configuration and metadata to make sure that they can be used to re-initialize the Solr service after the upgrade. The script performs a series of checks to make sure that:

- Required configuration files (such as `solr.xml`, `clusterstate.json`, and collection configuration sets) are present.
- The configuration files are compatible with the Solr version being upgraded to (Solr 7, in this case).

For example:

```
/opt/cloudera/cm/solr-upgrade/solr-upgrade.sh validate-metadata -c  
$HOME/cdh6-migrated-solr-config
```

If you have enabled Kerberos, specify your JAAS configuration file by appending `--jaas /path/to/solr-jaas.conf` to the command.

If the validation is successful, the script outputs a message similar to the following:

```
Validation successful for metadata in /home/solruser/cdh6-migrated-solr-config
```

If the validation fails, you can revisit the steps in [Migrate the Configuration](#) on page 83.

Test the Migrated Configuration on a CDH 6 Cluster

Although the configuration migration script addresses known incompatibilities, there might be incompatible configurations that are not detected by the script. Do not upgrade your production Cloudera Search environments to CDH 6 until you have tested the migrated configuration with CDH 6 on a testing or development cluster.

To test the migrated configuration on a CDH 6 cluster, you can either provision a new host in your upgraded Cloudera Manager 6 environment and add a CDH 6 cluster using that host, or you can continue to [Migrating Cloudera Search Configuration Before Upgrading to CDH 6](#) on page 79 and upgrade a development or test cluster to CDH 6. You can then use the CDH 6 cluster to test the migrated configuration to make sure that it works on CDH 6 before upgrading your production environment.

Copy the Migrated Configuration to the Upgrade Metadata Directory

After you have successfully migrated the configuration, and verified that the updated configuration works in CDH 6, you must copy it to the directory you designated as the **Upgrade Metadata Directory** in [Set Configuration Properties in Cloudera Manager](#) on page 82 (`/cdh6-solr-metadata/migrated-config` in this example), and change the ownership to the Solr service superuser. For example:

```
sudo mkdir -p /cdh6-solr-metadata/migrated-config
```

```
sudo cp -r $HOME/cdh6-migrated-solr-config/* /cdh6-solr-metadata/migrated-config
```

```
sudo chown -R solr:solr /cdh6-solr-metadata
```

If you have made any changes to the configuration after testing on a CDH 6 cluster, make sure that you copy the updated configuration from the CDH 6 cluster to the CDH 5 cluster you are upgrading.

Upgrade to CDH 6

After completing all of these procedures, upgrade to CDH following the regular process as documented in [Upgrading the CDH Cluster](#) on page 92. After the upgrade is complete, continue to [Re-Indexing Solr Collections After Upgrading to CDH 6](#) on page 110.

Migrating Apache Spark Before Upgrading to CDH 6

If you are upgrading to CDH 6 from CDH 5 and have a Spark service installed, there are several pre-upgrade steps you might need to take:

Remove Spark (Standalone) Service

If you have a **Spark (Standalone)** service in your cluster, remove it before starting the CDH upgrade:

1. Log in to the Cloudera Manager Admin Console.
2. Click the drop-down arrow next to the **Spark (Standalone)** service and select **Stop**.
3. Click the drop-down arrow next to the **Spark (Standalone)** service and select **Delete**.

Set Alternatives Priorities for Built-in CDH 5 Spark (1.6) and CDS 2



Important: CDH 6 does not support using multiple versions of Spark. Upgrading to CDH 6 disables the [CDS 2](#) parcel if it exists, and all Spark services will use the built-in Spark version ([2.2 in CDH 6.0.0](#)). For example, if you have two CDH 5 Spark (1.6) services, and two CDS 2 Spark (2.x) services, after upgrading to CDH 6, you will have four Spark services, all using the built-in CDH 6 Spark version.

Additionally, the command used to submit Spark 2 jobs in CDH 5 (`spark2-submit`) is removed in CDH 6, replaced by `spark-submit`. In a CDH 5 cluster with both the built-in Spark 1.6 service and a Spark 2 service, `spark-submit` is used with the Spark 1.6 service, and `spark2-submit` is used with the Spark 2 service. After upgrading to CDH 6, `spark-submit` uses the CDH built-in Spark 2 service, and `spark2-submit` does not work. Make sure to update any workflows that submit Spark jobs using these commands after upgrading to CDH 6.

If you are using both the built-in Spark 1.6 service in CDH 5 and a [CDS 2](#) parcel, and both services have **Gateway** roles on the same hosts, increase the alternatives priority of the service that you want to use as the default service after the upgrade.

To set the alternatives priority

1. Log in to the Cloudera Manager Admin Console.
2. Select the Cluster where the Spark services are running.
3. Select the Spark service that you want to use as the default service after upgrading.
4. Click the **Configuration** tab.
5. Search for the **Alternatives Priority** property.
6. Set the value higher than the **Alternatives Priority** for any other Spark service.
7. Click **Save Changes**.

Remove CDS 2 Version Higher than the CDH 6 Spark 2 Version



Important: The command used to submit Spark 2 jobs in CDH 5 (`spark2-submit`) is removed in CDH 6, replaced by `spark-submit`. In a CDH 5 cluster with both the built-in Spark 1.6 service and a Spark 2 service, `spark-submit` is used with the Spark 1.6 service, and `spark2-submit` is used with the Spark 2 service. After upgrading to CDH 6, `spark-submit` uses the CDH built-in Spark 2 service, and `spark2-submit` does not work. Make sure to update any workflows that submit Spark jobs using these commands after upgrading to CDH 6.

If you are using a [CDS 2](#) minor version higher than the version of Spark 2 included in the CDH 6 release you are upgrading to ([2.2 in CDH 6.0.0](#)), you must remove your Spark 2 services from Cloudera Manager. For the purpose of this evaluation,

you can ignore maintenance versions. For example, if the Spark 2 version in the CDH 6 version you are upgrading to is 2.2, and you are using any maintenance version of CDS 2.2.0, you do not need to remove your Spark 2 services from Cloudera Manager. They will be automatically converted to use the built-in CDH 6 Spark version, and the CDS parcel will be disabled.

Deleting a Spark service in Cloudera Manager does not delete the associated event logs from HDFS. The CDH 6 upgrade wizard installs Spark 2.2.

To remove the Spark service:

1. Log in to the Cloudera Manager Admin Console.
2. Select the Cluster where the Spark 2 service is running.
3. Click the drop-down arrow next to the Spark service and select **Stop**.
4. Click the drop-down arrow next to the Spark service and select **Delete**.
5. After upgrading to CDH 6, add the Spark service. For instructions, see [Adding a Service](#).

Migrating Apache HBase Before Upgrading to CDH 6

If you are upgrading to CDH 6 from CDH 5 and have the HBase service installed, there are several pre-upgrades steps you might need to take:

- [Remove PREFIX_TREE Data Block Encoding](#) on page 87
- [Upgrade Co-Processor Classes](#) on page 88
- [Check the HBase Related Upgrade Checkboxes](#) on page 88

Remove PREFIX_TREE Data Block Encoding

In CDH 6, HBase 2.0 is included. HBase 2.0 does not support PREFIX_TREE Data Block Encoding. Therefore, before upgrading to CDH 6 PREFIX_TREE Data Block Encoding must be changed to a supported encoding, otherwise HBase 2.0 fails to start.

If you already have a CDH 6 installation, you can ensure that none of your tables or snapshots use the PREFIX_TREE Data Block Encoding by running the following tools:

- `hbase pre-upgrade validate-dbe`
- `hbase pre-upgrade validate-hfile`

If you do not have a CDH 6 installation, you can still use these tools if you download and distribute the CDH 6 parcel:

1. Download and distribute parcels for target version of CDH 6 as described in [Download and Distribute Parcel Step 1-3](#).



Important: Do not activate the CDH 6.x parcel yet.

If the downloaded parcel version is higher than the current Cloudera Manager version the following error message displayed:

Error for parcel CDH-6.X.parcel : Parcel version 6.X is not supported by this version of Cloudera Manager. Upgrade Cloudera Manager to at least 6.X before using this version of the parcel.

2. Ignore the error message.
3. Run the pre-upgrade command using the CDH 6 parcel preferably on a HMaster host.

For example:

```
$ /opt/cloudera/parcels/CDH-6.1.1-1.cdh6.1.1.p0.875250/bin/hbase pre-upgrade validate-dbe
$ /opt/cloudera/parcels/CDH-6.1.1-1.cdh6.1.1.p0.875250/bin/hbase pre-upgrade validate-hfile
```

Upgrade Co-Processor Classes

External co-processors are not automatically upgraded. There are two ways to handle co-processor upgrade:

- Upgrade your co-processor jars manually before continuing the upgrade.
- Temporarily unset the co-processors and continue the upgrade.

Once they are manually upgraded, they can be reset.

Attempting to upgrade without upgrading the co-processor jars can result in unpredictable behaviour such as HBase role start failure, HBase role crashing, or even data corruption.

If you already have a CDH 6 installation, you can ensure that your co-processors are compatible with the upgrade by running the `hbase pre-upgrade validate-cp` tool.

If you do not have a CDH 6 installation, you can still use this tool if you download and distribute the CDH 6 parcel:

1. Download and distribute parcels for target version of CDH 6 as described in [Download and Distribute Parcel Step 1-3](#).



Important: Do not activate the CDH 6.x parcel yet.

If the downloaded parcel version is higher than the current Cloudera Manager version the following error message displayed:

Error for parcel CDH-6.X.parcel : Parcel version 6.X is not supported by this version of Cloudera Manager. Upgrade Cloudera Manager to at least 6.X before using this version of the parcel.

2. Ignore the error message.
3. Run the pre-upgrade command using the CDH 6 parcel preferably on a HMaster host.

For example, check for co-processor compatibility on master:

```
$ /opt/cloudera/parcels/CDH-6.1.1-1.cd6.1.1.p0.875250/bin/hbase pre-upgrade validate-cp -config
```

On regionserver:

```
$ /opt/cloudera/parcels/CDH-6.1.1-1.cd6.1.1.p0.875250/bin/hbase pre-upgrade validate-cp -table .*
```

Check the HBase Related Upgrade Checkboxes

When you are attempting to upgrade from a CDH 5 cluster to a CDH 6 cluster, checkboxes appear to ensure you have performed all the HBase related pre-upgrade migration steps.

The upgrade continues only if you check both of the following statements:

- Yes, I have run [HBase pre-upgrade checks](#).
- Yes, I have manually [upgraded the HBase co-processor classes](#).

- ⓘ HBase 2.0 does not support PREFIX_TREE Data Block Encoding. It must be changed to a supported encoding, otherwise HBase 2.0 fails. Run 'hbase pre-upgrade validate-dbe' and 'hbase pre-upgrade validate-hfile' to be sure none of your tables / snapshots use it.
 - ☐ Yes, I have run HBase pre-upgrade checks.
- ⓘ External HBase co-processors are not automatically upgraded. If your co-processor jars have been manually upgraded, you can continue with the upgrade. Otherwise, you should temporarily unset the co-processors. They can be reset later after they have been manually upgraded. Attempting to upgrade the co-processor jars can result in unpredictable behavior such as HBase roles failing to start, or crashing, or even data corruption. To check co-processor compatibility, please run 'hbase pre-upgrade validate-cp'.
 - ☐ Yes, I have manually upgraded the HBase co-processor classes.

Installing Dependencies for Hue

This page is only required when upgrading to CDH 6 or higher and one of the following is true:

- Hue is installed on a RHEL 6 or compatible host.
- Hue is using PostgreSQL.

(RHEL 6 Compatible Only) Install Python 2.7 on Hue Hosts

Hue in CDH 6 requires Python 2.7, which is included by default in RHEL 7 compatible operating systems (OSes).

RHEL 6 compatible OSes include Python 2.6. You must install Python 2.7 on all Hue hosts before installing or upgrading to Cloudera Enterprise 6:

RHEL 6

1. Make sure that you have access to the Software Collections Library. For more information, see the Red Hat knowledge base article, [How to use Red Hat Software Collections \(RHSC\) or Red Hat Developer Toolset \(DTS\)?](#).
2. Install Python 2.7:

```
sudo yum install python27
```

3. Verify that Python 2.7 is installed:

```
source /opt/rh/python27/enable
python --version
```

CentOS 6

1. Enable the Software Collections Library:

```
sudo yum install centos-release-scl
```

2. Install the Software Collections utilities:

```
sudo yum install scl-utils
```

3. Install Python 2.7:

```
sudo yum install python27
```

4. Verify that Python 2.7 is installed:

```
source /opt/rh/python27/enable
python --version
```

Oracle Linux 6

1. Download the Software Collections Library repository:

```
sudo wget -O /etc/yum.repos.d/public-yum-ol6.repo
http://yum.oracle.com/public-yum-ol6.repo
```

2. Edit `/etc/yum.repos.d/public-yum-ol6.repo` and make sure that `enabled` is set to 1, as follows:

```
[ol6_software_collections]
name=Software Collection Library release 3.0 packages for Oracle Linux 6 (x86_64)
baseurl=http://yum.oracle.com/repo/OracleLinux/OL6/SoftwareCollections/x86_64/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1
```

For more information, see [Installing the Software Collection Library Utility From the Oracle Linux Yum Server](#) in the Oracle documentation.

3. Install the Software Collections utilities:

```
sudo yum install scl-utils
```

4. Install Python 2.7:

```
sudo yum install python27
```

5. Verify that Python 2.7 is installed:

```
source /opt/rh/python27/enable
python --version
```

Installing the `psycopg2` Python Package

Hue in CDH 6 requires version 2.5.4 or higher of the `psycopg2` Python package for connecting to a PostgreSQL database. The `psycopg2` package is automatically installed as a dependency of Cloudera Manager Agent, but the version installed is often lower than 2.5.4.

If you are installing or upgrading to CDH 6 and using PostgreSQL for the Hue database, you must install `psycopg2` 2.5.4 or higher on all Hue hosts as follows. These examples install version 2.7.5 (2.6.2 for RHEL 6):

RHEL 7 Compatible

1. Install the `python-pip` package:

```
sudo yum install python-pip
```

2. Install `psycopg2` 2.7.5 using `pip`:

```
sudo pip install psycopg2==2.7.5 --ignore-installed
```

RHEL 6 Compatible

1. Make sure that you have [installed Python 2.7](#). You can verify this by running the following commands:

```
source /opt/rh/python27/enable
python --version
```

2. Install the `python-pip` package:

```
sudo yum install python-pip
```

3. Install the postgresql-devel package:

```
sudo yum install postgresql-devel
```

4. Install the gcc* packages:

```
sudo yum install gcc*
```

5. Install psycopg2 2.6.2 using pip:

```
sudo bash -c "source /opt/rh/python27/enable; pip install psycopg2==2.6.2 --ignore-installed"
```

Ubuntu / Debian**1. Install the python-pip package:**

```
sudo apt-get install python-pip
```

2. Install psycopg2 2.7.5 using pip:

```
sudo pip install psycopg2==2.7.5 --ignore-installed
```

SLES 12

Install the python-psycopg2 package:

```
sudo zypper install python-psycopg2
```

Pre-Upgrade Migration Steps for Upgrading Key Trustee KMS to CDH 6

If you are upgrading from CDH 5.x and have the Key Trustee KMS service installed, you must first activate the KT KMS 6.x service.

- **Pre-Upgrade Steps for Parcels**

You must upgrade Key Trustee KMS *before* upgrading CDH. For details on how to upgrade Key Trustee KMS, refer to [Upgrading Key Trustee KMS Using Parcels](#) on page 171.

- **Pre-Upgrade Steps for Packages**

You must upgrade Key Trustee KMS *before* upgrading CDH. For details on how to upgrade Key Trustee KMS, refer to [Upgrading Key Trustee KMS Using Packages](#) on page 171.

Pre-Upgrade Migration Steps for Upgrading HSM KMS to CDH 6

If you are upgrading from CDH 5.x and have the HSM KMS service installed, you must first activate the HSM KMS 6.x service.

- **Pre-Upgrade Steps for Parcels**

You must upgrade HSM KMS *before* upgrading CDH. For details on how to upgrade HSM KMS, refer to [Upgrading HSM KMS Using Parcels](#) on page 171.

- **Pre-Upgrade Steps for Packages**

You must upgrade HSM KMS *before* upgrading CDH. For details on how to upgrade HSM KMS, refer to [Upgrading HSM KMS Using Packages](#) on page 172.

Upgrading the CDH Cluster

The version of CDH you can upgrade to [depends on the version of Cloudera Manager](#) that is managing the cluster. You may need to [upgrade Cloudera Manager](#) before upgrading CDH.

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)



Warning: Redhat 5 is not a supported operating system for CDH 6.x.



Warning: SLES 11 is not a supported operating system for CDH 6.x.



Warning: Ubuntu 14 and lower are not supported operating systems for CDH 6.x.



Warning: Debian is not a supported operating system for CDH 6.x.



Warning: Ubuntu 12 is not a supported operating systems for CDH 5.16 and higher.



Important: CDH 6.0 supports upgrading from CDH 5.7 or higher, up to 5.14.x, but **NOT** from CDH 5.15.x or higher.



Warning: Kafka in CDH 6.2.0 is based on Apache Kafka 2.1.0, which contains a change to the internal schema used to store consumer offsets. As a result of this change, downgrading Kafka to a version lower than CDH 6.2.0 is **NOT** possible once Kafka has been upgraded to CDH 6.2.0 or higher.



Warning: Backing up data using Cloudera Manager Backup and Disaster Recovery (BDR) does not work when backing up a cluster from Cloudera Manager 5.13 and lower to CDH 6.0 or higher.



Warning: Cloudera Data Science Workbench is not supported with CDH 6.0.x. If you proceed with the upgrade, you will be asked to remove the CDSW service from your cluster. Cloudera Data Science Workbench 1.5 (and higher) is supported with CDH 6.1 (and higher).

**Important:**

- This procedure is for CDH clusters only. [Follow this procedure](#) to upgrade a Key Trustee Server Cluster and this can be carried out independently.
- The embedded PostgreSQL database is **NOT supported** in production environments.
- If you use the Solr Search service in your cluster, there are significant manual steps you must follow both before and after upgrading CDH. See [Migrating Cloudera Search Configuration Before Upgrading to CDH 6](#) on page 79.
- The following services are no longer supported as of Enterprise 6.0.0:
 - Sqoop 2
 - MapReduce 1
 - Spark 1.6
 - Record Service
- Running Apache Accumulo on top of a CDH 6.0.0 cluster is not currently supported. If you try to upgrade to CDH 6.0.0 you will be asked to remove the Accumulo service from your cluster. Running Accumulo on top of CDH 6 will be supported in a future release.
- The minor version of Cloudera Manager you use to perform the upgrade must be equal to or greater than the CDH minor version. To upgrade Cloudera Manager, see [Upgrading Cloudera Manager](#) on page 38.
 - Supported:
 - Cloudera Manager 6.0.0 and CDH 5.14.0
 - Cloudera Manager 5.14.0 and CDH 5.13.0
 - Cloudera Manager 5.13.1 and CDH 5.13.3
 - Not Supported:
 - Cloudera Manager 5.14.0 and CDH 6.0.0
 - Cloudera Manager 5.12 and CDH 5.13
 - Cloudera Manager 6.0.0 and CDH 5.6

**Note:**

When upgrading CDH using **Rolling Restart** (Minor Upgrade only):

- Automatic failover does not affect the rolling restart operation.
- After the upgrade has completed, do not remove the old parcels if there are MapReduce or Spark jobs currently running. These jobs still use the old parcels and must be restarted in order to use the newly upgraded parcel.
- Ensure that Oozie jobs are idempotent.
- Do not use Oozie Shell Actions to run Hadoop-related commands.
- Rolling upgrade of Spark Streaming jobs is not supported. Restart the streaming job once the upgrade is complete, so that the newly deployed version starts being used.
- Runtime libraries must be packaged as part of the Spark application.
- You must use the distributed cache to propagate the job configuration files from the client gateway machines.
- Do not build "uber" or "fat" JAR files that contain third-party dependencies or CDH classes as these can conflict with the classes that Yarn, Oozie, and other services automatically add to the CLASSPATH.
- Build your Spark applications without bundling CDH JARs.



Note: If you are upgrading to a *maintenance* version of CDH, skip any steps that are labeled **[Not required for CDH maintenance release upgrades.]**.


After you complete the steps to [prepare your CDH upgrade](#) and [backup CDH components](#), continue with the following upgrade steps:

Back Up Cloudera Manager

Before you upgrade a CDH cluster, back up Cloudera Manager. Even if you just backed up Cloudera Manager before an upgrade, you should now back up your upgraded Cloudera Manager deployment. See [Backing Up Cloudera Manager](#) on page 41.

Enter Maintenance Mode

To avoid unnecessary alerts during the upgrade process, enter maintenance mode on your cluster before you start the upgrade. Entering maintenance mode stops email alerts and SNMP traps from being sent, but does not stop checks and configuration validations. Be sure to exit maintenance mode when you have finished the upgrade to re-enable Cloudera Manager alerts. [More Information](#).

On the **Home > Status** tab, click  next to the cluster name and select **Enter Maintenance Mode**.

Complete Pre-Upgrade Migration Steps

Complete the following steps when upgrading from CDH 5.x to CDH 6.x.

- **YARN**

Decommission and recommission the YARN NodeManagers but do not start the NodeManagers.

A decommission is required so that the NodeManagers stop accepting new containers, kill any running containers, and then shutdown.

1. Ensure that new applications, such as MapReduce or Spark applications, will not be submitted to the cluster until the upgrade is complete.
2. In the Cloudera Manager Admin Console, navigate to the YARN service for the cluster you are upgrading.
3. On the **Instances** tab, select all the **NodeManager** roles. This can be done by filtering for the roles under **Role Type**.
4. Click **Actions for Selected (number) > Decommission**.

If the cluster runs CDH 5.9 or higher and is managed by Cloudera Manager 5.9 or higher, and you configured graceful decommission, the countdown for the timeout starts.

A **Graceful Decommission** provides a timeout before starting the decommission process. The timeout creates a window of time to drain already running workloads from the system and allow them to run to completion. Search for the **Node Manager Graceful Decommission Timeout** field on the **Configuration** tab for the YARN service, and set the property to a value greater than 0 to create a timeout.

5. Wait for the decommissioning to complete. The NodeManager **State** is **Stopped** and the **Commission State** is **Decommissioned** when decommissioning completes for each NodeManager.
6. With all the NodeManagers still selected, click **Actions for Selected (number) > Recommission**.



Important: Do not start the NodeManagers.

- **Hive**

There are changes to query syntax, DDL syntax, and the Hive API. You might need to edit the HiveQL code in your application workloads before upgrading.

See [Incompatible Changes for Apache Hive/Hive on Spark/HCatalog](#).

- **Pig**

DataFu is no longer supported. Your Pig scripts will require modification for use with CDH 6.x.

See [Incompatible Changes for Apache Pig](#).

- **Sentry**

If your cluster uses Sentry policy file authorization, you must migrate the policy files to the database-backed Sentry service before you upgrade to CDH 6.

See [Migrating from Sentry Policy Files to the Sentry Service](#) on page 78.

- **Cloudera Search**

If your cluster uses Cloudera Search, you must migrate the configuration to Apache Solr 7.

See [Migrating Cloudera Search Configuration Before Upgrading to CDH 6](#) on page 79.

- **Spark**

If your cluster uses Spark or Spark Standalone, there are several steps you must perform to ensure that the correct version is installed.

See [Migrating Apache Spark Before Upgrading to CDH 6](#) on page 86.

- **Kafka**

In CDH 5.x, Kafka was delivered as a separate parcel and could be installed along with CDH 5.x using Cloudera Manager. Starting with CDH 6.0, Kafka is part of the CDH distribution and is deployed as part of the CDH 6.x parcel.

1. Explicitly set the Kafka protocol version to match what's being used currently among the brokers and clients. Update server.properties on all brokers as follows:

- a. Log in to the Cloudera Manager Admin Console
- b. Choose the Kafka service.
- c. Click **Configuration**.
- d. Use the Search field to find the **Kafka Broker Advanced Configuration Snippet (Safety Valve) for kafka.propertiesconfiguration** property.
- e. Add the following properties to the snippet:

- `inter.broker.protocol.version = current_Kafka_version`
- `log.message.format.version = current_Kafka_version`

Make sure you enter full Kafka version numbers with three values, such as 0.10.0. Otherwise, you will see an error message similar to the following:

```
2018-06-14 14:25:47,818 FATAL kafka.Kafka$:
java.lang.IllegalArgumentException: Version `0.10` is not a valid version
    at kafka.api.ApiVersion$$anonfun$apply$1.apply(ApiVersion.scala:72)
    at kafka.api.ApiVersion$$anonfun$apply$1.apply(ApiVersion.scala:72)
    at scala.collection.MapLike$class.getOrElse(MapLike.scala:128)
```

2. Save your changes. The information is automatically copied to each broker.

- **HBase**

See [Migrating Apache HBase Before Upgrading to CDH 6](#) on page 87.

- **Hue**

See [Installing Dependencies for Hue](#) on page 89.

- **Key Trustee KMS**

See [Pre-Upgrade Migration Steps for Upgrading Key Trustee KMS to CDH 6](#) on page 91.

- **HSM KMS**

See [Pre-Upgrade Migration Steps for Upgrading HSM KMS to CDH 6](#) on page 91.

Establish Access to the Software

If you are installing CDH using packages:

- If the Cloudera Manager hosts have internet access, you can use the publicly available repositories from <https://archive.cloudera.com>. Modify the sample repo files below for your operating system and version.
- If the Cloudera Manager hosts *do not* have internet access, configure [a local package repository](#) hosted on your network. For example: <http://MyWebServer:1234/cloudera-repos>
- Replace archive.cloudera.com in the sample repo files below with the URL for your local repository.

Log in to each host in the cluster using ssh and perform the following steps:

1. Back up the existing repository directory.

RHEL / CentOS

```
sudo cp -rpf /etc/yum.repos.d $HOME/yum.repos.d-`date +%F`-CM-CDH
```

SLES

```
sudo cp -rpf /etc/zypp/repos.d $HOME/repos.d-`date +%F`CM-CDH
```

Debian / Ubuntu

```
sudo cp -rpf /etc/apt/sources.list.d $HOME/sources.list.d-`date +%F`-CM-CDH
```

2. Remove any older files in the existing repository directory:

RHEL / CentOS

```
sudo rm /etc/yum.repos.d/cloudera*cdh.repo*
```

SLES

```
sudo rm /etc/zypp/repos.d/cloudera*cdh.repo*
```

Debian / Ubuntu

```
sudo rm /etc/apt/sources.list.d/cloudera*cdh.list*
```

3. **RHEL / CentOS**

Create a file named `/etc/yum.repos.d/cloudera-cdh.repo` with the following content:

```
[cdh]
# Packages for CDH
name=CDH
baseurl=https://archive.cloudera.com/cdh5/redhat/7/x86_64/cdh/5.15
gpgkey=https://archive.cloudera.com/cdh5/redhat/7/x86_64/cdh/RPM-GPG-KEY-cloudera
gpgcheck=1
```


SLES

Create a file named `/etc/zypp/repos.d/cloudera-cdh.repo` with the following content:

```
[cdh]
# Packages for CDH
name=CDH
baseurl=https://archive.cloudera.com/cdh5/sles/12/x86_64/cdh/5.15
gpgkey=https://archive.cloudera.com/cdh5/sles/12/x86_64/cm/RPM-GPG-KEY-cloudera
gpgcheck=1
```

Debian / Ubuntu

Create a file named `/etc/apt/sources.list.d/cloudera-cdh.list` with the following content:

```
# Packages for CDH
deb https://archive.cloudera.com/cdh5/debian/jessie/amd64/cdh/ jessie-cdh5.15 contrib
deb-src https://archive.cloudera.com/cdh5/debian/jessie/amd64/cdh/ jessie-cdh5.15 contrib
```

```
sudo apt-get update
```

The repository file, as created, specifies an upgrade to the *most recent* maintenance release of the specified minor release. If you would like to use a *specific* maintenance version, for example 5.15.1, replace 5.15 with 5.15.1 in the generated repository file shown above.

Run Hue Document Cleanup

If your cluster uses Hue, perform the following steps (not required for maintenance releases). These steps clean up the database tables used by Hue and can help improve performance after an upgrade.

1. Backup the Hue Database.
2. Check the size of the `desktop_document`, `desktop_document2`, `oozie_job`, `beeswax_session`, `beeswax_savedquery` and `beeswax_queryhistory` tables to have a reference point. If any of these have more than 100k rows, run the cleanup.

```
select count * from desktop_document;
select count * from desktop_document2;
select count * from beeswax_session;
select count * from beeswax_savedquery;
select count * from beeswax_queryhistory;
select count * from oozie_job;
```

3. Pick a node with a running Hue instance, the script requires Hue to be running and uses the running configuration of Hue to run. Download the `hue_scripts` repo to any Hue node by following the `git` or `wget` steps. These scripts are a set of libraries and commands, the **ENTIRE** repo is required.

```
wget -qO- -O /tmp/hue_scripts.zip
https://github.com/cmconner156/hue_scripts/archive/master.zip && unzip -d /tmp
/tmp/hue_scripts.zip
mv /tmp/hue_scripts-master /opt/cloudera/hue_scripts
```

4. Change the permissions on the script runner to make it runnable:

```
chmod 700 /opt/cloudera/hue_scripts/script_runner
```

5. Run the script as root on that node, the command is all one line, `DESKTOP_DEBUG=True` is setup for the environment of the single run of this command so you don't have to tail a log:

```
DESKTOP_DEBUG=True /opt/cloudera/hue_scripts/script_runner hue_desktop_document_cleanup
--keep-days 90
```

6. If you included the `DESKTOP_DEBUG` above, logging will be in the console. Otherwise check

`/var/log/hue/hue_desktop_document_cleanup.log`.

- **Note:** The first run typically takes around 1 minute per 1000 entries in each table (but can take much longer depending on the size of the tables).

7. Check the size of the `desktop_document`, `desktop_document2`, `oozie_job`, `beeswax_session`, `beeswax_savedquery` and `beeswax_queryhistory` tables and confirm they are now smaller.

```
select count(*) from desktop_document;
select count(*) from desktop_document2;
select count(*) from beeswax_session;
select count(*) from beeswax_savedquery;
select count(*) from beeswax_queryhistory;
select count(*) from oozie_job;
```

8. If any of the tables are still above 100k in size, run the command again, keeping less days:

```
--keep-days 60 or --keep-days 30
```

Check Oracle Database Initialization

If your cluster uses Oracle for any databases, before upgrading from CDH 5 to CDH 6, check the value of the `COMPATIBLE` initialization parameter in the Oracle Database using the following SQL query:

```
SELECT name, value FROM v$parameter WHERE name = 'compatible'
```

The default value is 12.2.0. If the parameter has a different value, you can set it to the default as shown in the [Oracle Database Upgrade Guide](#).



Note: Before resetting the `COMPATIBLE` initialization parameter to its default value, make sure you consider the effects of this change can have on your system.

Stop the Cluster

If you are installing CDH using packages:

Stop the cluster before proceeding to upgrade CDH using packages:

1. Open the Cloudera Manager Admin Console.
2. Click the drop-down list next to the cluster name and select **Stop**.

Install CDH Packages

If you are installing CDH using packages:

1. Log in to each host in the cluster using `ssh`.
2. Run the following command:

RHEL / CentOS

CDH 5 upgrade:

```
sudo yum clean all
```

```
sudo yum install avro-tools crunch flume-ng hadoop-hdfs-fuse hadoop-httpfs hadoop-kms
hbase hbase-solr hive-hbase hive-webhcat hue-beeswax hue-hbase hue-impala hue-pig
hue-plugins hue-rdbms hue-search hue-spark hue-sqoop hue-zookeeper impala impala-shell
kite llama mahout oozie parquet pig pig-udf-datafu search sentry solr solr-mapreduce
spark-python sqoop sqoop2 whirr zookeeper
```

CDH 6 upgrade:

```
sudo yum clean all
```

```
sudo yum remove hadoop-0.20\* hue-\* crunch llama mahout sqoop2 whirr sqoop2-client
```

```
sudo yum install avro-tools flume-ng hadoop-hdfs-fuse hadoop-hdfs-nfs3 hadoop-httpfs
hadoop-kms hbase-solr hive-hbase hive-webhcat hue impala impala-shell kafka kite kudu
oozie pig search sentry sentry-hdfs-plugin solr-crunch solr-mapreduce spark-core
spark-python sqoop zookeeper parquet hbase solr
```

SLES**CDH 5 upgrade:**

```
sudo zypper clean --all
```

```
sudo zypper install avro-tools crunch flume-ng hadoop-hdfs-fuse hadoop-httpfs hadoop-kms
hbase hbase-solr hive-hbase hive-webhcat hue-beeswax hue-hbase hue-impala hue-pig
hue-plugins hue-rdbms hue-search hue-spark hue-sqoop hue-zookeeper impala impala-shell
kite llama mahout oozie parquet pig pig-udf-datafu search sentry solr solr-mapreduce
spark-python sqoop sqoop2 whirr zookeeper
```

CDH 6 upgrade:

```
sudo zypper clean --all
```

```
sudo zypper remove hadoop-0.20\* hue-\* crunch llama mahout sqoop2 whirr sqoop2-client
```

```
sudo zypper install avro-tools flume-ng hadoop-hdfs-fuse hadoop-hdfs-nfs3 hadoop-httpfs
hadoop-kms hbase-solr hive-hbase hive-webhcat hue impala impala-shell kafka kite kudu
oozie pig search sentry sentry-hdfs-plugin solr-crunch solr-mapreduce spark-core
spark-python sqoop zookeeper parquet hbase solr
```

Debian / Ubuntu**CDH 5 upgrade:**

```
sudo apt-get update
sudo apt-get install avro-tools crunch flume-ng hadoop-hdfs-fuse hadoop-httpfs hadoop-kms
hbase hbase-solr hive-hbase hive-webhcat hue-beeswax hue-hbase hue-impala hue-pig
hue-plugins hue-rdbms hue-search hue-spark hue-sqoop hue-zookeeper impala impala-shell
kite llama mahout oozie parquet pig pig-udf-datafu search sentry solr solr-mapreduce
spark-python sqoop sqoop2 whirr zookeeper
```

CDH 6 upgrade:

```
sudo apt-get update
sudo apt-get remove hadoop-0.20\* crunch llama mahout sqoop2 whirr sqoop2-client
```

```
sudo apt-get update
sudo apt-get install avro-tools flume-ng hadoop-hdfs-fuse hadoop-hdfs-nfs3 hadoop-httpfs
hadoop-kms hbase-solr hive-hbase hive-webhcat hue impala impala-shell kafka kite kudu
oozie pig search sentry sentry-hdfs-plugin solr-crunch solr-mapreduce spark-core
spark-python sqoop zookeeper parquet hbase solr
```

3. Restart the Cloudera Manager Agent.**RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher**

```
sudo systemctl restart cloudera-scm-agent
```

If the agent starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent restart
```

You should see the following:

```
Starting cloudera-scm-agent: [ OK ]
```

Download and Distribute Parcels

1. Log in to the Cloudera Manager Admin Console.
2. Click **Hosts > Parcels**. The **Parcels** page displays.
3. Update the Parcel Repository for CDH using the following remote parcel repository URL:

```
https://archive.cloudera.com/cdh5/parcels/5.15/
```

```
https://archive.cloudera.com/cdh6/6.0.0/parcels/
```

- a. Click the **Configuration** button.
 - b. In the **Remote Parcel Repository URLs** section, Click the **+** icon to add the parcel URL above. Click **Save Changes**. See [Parcel Configuration Settings](#) for more information.
 - c. Locate the row in the table that contains the new CDH parcel and click the **Download** button.
 - d. After the parcel is downloaded, click the **Distribute** button.
4. If your cluster has **GPLEXTRAS** installed, update the version of the GPLEXTRAS parcel to match the CDH version using the following remote parcel repository URL:

```
https://archive.cloudera.com/gplextras5/parcels/5.15/
```

```
https://archive.cloudera.com/gplextras6/6.0.0/parcels/
```

- a. Click the **Configuration** button.
 - b. In the **Remote Parcel Repository URLs** section, Click the **+** icon to add the parcel URL above. Click **Save Changes**. See [Parcel Configuration Settings](#) for more information.
 - c. Locate the row in the table that contains the new CDH parcel and click the **Download** button.
 - d. After the parcel is downloaded, click the **Distribute** button.
5. If your cluster has **Spark 2.0 or Spark 2.1** installed and you want to upgrade to CDH 5.13 or higher, you must [download](#) and install **Spark 2.1 release 2** or later.

To install these versions of Spark, do the following before running the CDH Upgrade Wizard:

1. Install the Custom Service Descriptor (CSD) file. See

- [Installing Spark 2.1](#) OR
- [Installing Spark 2.2](#)



Note:

Spark 2.2 requires that JDK 1.8 be deployed throughout the cluster. JDK 1.7 is not supported for Spark 2.2.

See [Upgrading the JDK](#) on page 22.

2. Download, distribute, and activate the Parcel for the version of Spark that you are installing:


- [Spark 2.1 release 2](#): The parcel name includes "cloudera2" in its name.
- [Spark 2.2 release 1](#): The parcel name includes "cloudera1" in its name.
- [Spark 2.2 release 2](#): The parcel name includes "cloudera2" in its name.

See [Managing Parcels](#).


6. If your cluster has **Kudu 1.4.0 or lower** installed and you want to upgrade to CDH 5.13 or higher, deactivate the existing Kudu parcel. Starting with Kudu 1.5.0 / CDH 5.13, Kudu is part of the CDH parcel and does not need to be installed separately.
7. After all the parcels are distributed, click on the **Upgrade** button next to the chosen CDH. The chosen CDH should be selected automatically.

Run the Upgrade CDH Wizard

1. If you are using packages, or did not choose **Upgrade** from the parcels page, You can get to the **Upgrade CDH**

page from the **Home > Status** tab, click  next to the cluster name and select **Upgrade Cluster**.

Select the previously download/distributed CDH version. If no qualifying CDH parcels are pre-listed, or you want to upgrade to a different version of CDH:

1. Click the **Remote Parcel Repository URLs** link and add the appropriate parcel URL. See [Parcel Configuration Settings](#) for more information.
2. Click the Cloudera Manager logo to return to the Home page.
3. From the **Home > Status** tab, click  next to the cluster name and select **Upgrade Cluster**.

If you were previously using packages and would like to switch to using parcels, select **Use Parcels**.

2. Cloudera Manager 5.14 and lower:

1. In the **Choose CDH Version (Parcels)** section, select the CDH version that you want to upgrade to.
2. Click **Continue**.

A page displays the version you are upgrading to and asks you to confirm that you have completed some additional steps.
3. Click **Yes, I have performed these steps**.
4. Click **Continue**.
5. Cloudera Manager verifies that the agents are responsive and that the correct software is installed. When you see the **No Errors Found** message, click **Continue**.

The selected parcels are downloaded, distributed, and unpacked.
6. Click **Continue**.

The [Host Inspector](#) runs. Examine the output and correct any reported errors.

Cloudera Manager 5.15 and higher:

1. In the **Upgrade to CDH Version** drop-down list, select the version of CDH you want to upgrade to.

The Upgrade Wizard performs some checks on configurations, health, and compatibility and reports the results. Fix any reported issues before continuing.
2. Click **Run Host Inspector**.

The [Host Inspector](#) runs. Click **Show Inspector Results** to view the Host Inspector report (opens in a new browser tab). Fix any reported issues before continuing.
3. Click **Run Service Inspector**. Click **Show Inspector Results** to view the output of the **Service Inspector** command (opens in a new browser tab). Fix any reported issues before continuing.

4. Read the notices for steps you must complete before upgrading, select **Yes, I have performed these steps.** ... after completing the steps, and click **Continue**.

The selected parcels are downloaded, distributed, and unpacked. The **Continue** button turns blue when this process finishes.

3. If you have a parcel that works with the existing CDH version, the Upgrade Wizard may display a message that this parcel conflicts with the new CDH version.
 - a. Configure and download the newer version of this parcel before proceeding.
 - a. Open the Cloudera Manager Admin Console from another browser tab, go to the parcels page, and configure the remote parcel repository for the newer version of this parcel.
 - b. Download and distribute the newer version of this parcel.
 - b. Click the **Run All Checks Again** button.
 - c. Select the option to resolve the conflicts automatically.
 - d. Cloudera Manager deactivates the old version of the parcel, activates the new version and verifies that all hosts have the correct software installed.
4. Click **Continue**.

The **Choose Upgrade Procedure** screen displays. Select the upgrade procedure from the following options:

- **Rolling Restart**



Note: This option is only available if you have [enabled high availability for HDFS](#), the cluster uses an Enterprise license, and you are performing a minor or maintenance upgrade.

Cloudera Manager upgrades services and performs a rolling restart. The **Rolling Restart** dialog box displays the impact of the restart on various services. Services that do not support rolling restart undergo a normal restart, and are not available during the restart process.

Configure the following parameters for the rolling restart (optional):

Roles to include

Select which roles to restart as part of the rolling restart.

Batch Size

Number of roles to include in a batch. Cloudera Manager restarts the worker roles rack-by-rack, in alphabetical order, and within each rack, hosts are restarted in alphabetical order. If you use the default replication factor of 3, Hadoop tries to keep the replicas on at least 2 different racks. So if you have multiple racks, you can use a higher batch size than the default 1. However, using a batch size that is too high means that fewer worker roles are active at any time during the upgrade, which can cause temporary performance degradation. If you are using a single rack, restart *one worker node at a time* to ensure data availability during upgrade.

Advanced Options > Sleep between batches

Amount of time Cloudera Manager waits before starting the next batch. Applies only to services with worker roles.

Advanced Options > Failed threshold

The number of *batch* failures that cause the entire rolling restart to fail. For example, if you have a very large cluster, you can use this option to allow some failures when you are sure that the cluster will still be functional while some worker roles are down.

Click the **Rolling Restart** button when you are ready to restart the cluster.

- **Full Cluster Restart**

Cloudera Manager performs all service upgrades and restarts the cluster.

- **Manual Upgrade**

Cloudera Manager configures the cluster to the specified CDH version but performs no upgrades or service restarts. Manually upgrading is difficult and for advanced users only. Manual upgrades allow you to selectively stop and restart services to prevent or mitigate downtime for services or clusters where rolling restarts are not available.

To perform a manual upgrade: See [Upgrading CDH Manually after an Upgrade Failure](#) on page 124 for the required steps.

5. Click **Continue**.

The **Upgrade Cluster Command** screen displays the result of the commands run by the wizard as it shuts down all services, activates the new parcels, upgrades services, deploys client configuration files, and restarts services and performs a rolling restart of the services that support it.

If any of the steps fail, correct any reported errors and click the **Resume** button. Cloudera Manager will skip restarting roles that have already successfully restarted. Alternately, return to the **Home > Status** tab and then perform the steps in [Upgrading CDH Manually after an Upgrade Failure](#) on page 124.



Note: If Cloudera Manager detects a failure while upgrading CDH, Cloudera Manager displays a dialog box where you can create a diagnostic bundle to send to Cloudera Support so they can help you recover from the failure. The cluster name and time duration fields are pre-populated to capture the correct data.

6. Click **Continue**.

If your cluster was previously installed or upgraded using *packages*, the wizard may indicate that some services cannot start because their parcels are not available. To download the required parcels:

1. In another browser tab, open the Cloudera Manager Admin Console.
2. Select **Hosts > Parcels**.
3. Locate the row containing the missing parcel and click the button to **Download, Distribute**, and then **Activate** the parcel.
4. Return to the upgrade wizard and click the **Resume** button.

The Upgrade Wizard continues upgrading the cluster.

7. Click **Finish** to return to the Home page.

Remove the Previous CDH Version Packages and Refresh Symlinks

[Not required for CDH maintenance release upgrades.]

If your previous installation of CDH was done using packages, remove those packages on all hosts where you installed the parcels and refresh the symlinks so that clients will run the new software versions.

Skip this step if your previous installation or upgrade used parcels.

1. If your Hue service uses the embedded SQLite database, back up `/var/lib/hue/desktop.db` to a location that is not `/var/lib/hue` because this directory is removed when the packages are removed.
2. Uninstall the CDH packages on each host:

Not including Impala and Search

RHEL / CentOS

```
sudo yum remove bigtop-utils bigtop-jsvc bigtop-tomcat 'hue-*' sqoop2-client
```

SLES

```
sudo zypper remove bigtop-utils bigtop-jsvc bigtop-tomcat 'hue-*' sqoop2-client
```

Debian / Ubuntu

```
sudo apt-get purge bigtop-utils bigtop-jsvc bigtop-tomcat 'hue-*' sqoop2-client
```

Including Impala and Search

RHEL / CentOS

```
sudo yum remove 'bigtop-*' 'hue-*' impala-shell solr-server sqoop2-client hbase-solr-doc  
avro-libs crunch-doc avro-doc solr-doc
```

SLES

```
sudo zypper remove 'bigtop-*' 'hue-*' impala-shell solr-server sqoop2-client  
hbase-solr-doc avro-libs crunch-doc avro-doc solr-doc
```

Debian / Ubuntu

```
sudo apt-get purge 'bigtop-*' 'hue-*' impala-shell solr-server sqoop2-client  
hbase-solr-doc avro-libs crunch-doc avro-doc solr-doc
```

3. Restart all the Cloudera Manager Agents to force an update of the symlinks to point to the newly installed components on each host.
4. If your Hue service uses the embedded SQLite database, restore the database you backed up:
 - a. Stop the Hue service.
 - b. Copy the backup from the temporary location to the newly created Hue database directory, `/var/lib/hue`.
 - c. Start the Hue service.

Complete the Cloudera Search Upgrade

Required for upgrades to 6.0.0 only. If the cluster you are upgrading has Cloudera Search (Solr) enabled, perform the following steps:

1. Log in to the Cloudera Manager Admin Console.
2. Go to the Solr service page.
3. Stop the Solr service and dependent services. Click **Actions > Stop**.
4. Click **Actions > Reinitialize Solr State for Upgrade**.
5. Click **Actions > Bootstrap Solr Configuration**.
6. Start the Solr and dependent services. Click **Actions > Start**.
7. Click **Actions > Bootstrap Solr Collections**.

Finalize the HDFS Upgrade

Follow the steps in this section if you are upgrading:

- CDH 5.0 or 5.1 to 5.2 or higher
- CDH 5.2 or 5.3 to 5.4 or higher

To determine if you can finalize the upgrade, run important workloads and ensure that they are successful. After you have finalized the upgrade, you cannot roll back to a previous version of HDFS without using backups. Verifying that you are ready to finalize the upgrade can take a long time.

Make sure you have enough free disk space, keeping in mind that the following behavior continues until the upgrade is finalized:

- Deleting files does not free up disk space.
- Using the balancer causes all moved replicas to be duplicated.
- All on-disk data representing the NameNodes metadata is retained, which could more than double the amount of space required on the NameNode and JournalNode disks.

If you have Enabled high availability for HDFS, and you have performed a rolling upgrade:

1. Go to the HDFS service.
2. Select **Actions** > **Finalize Rolling Upgrade** and click **Finalize Rolling Upgrade** to confirm.

If you have not performed a rolling upgrade:

1. Go to the HDFS service.
2. Click the **Instances** tab.
3. Click the link for the **NameNode** instance. If you have enabled high availability for HDFS, click the link labeled **NameNode (Active)**.

The NameNode instance page displays.

4. Select **Actions** > **Finalize Metadata Upgrade** and click **Finalize Metadata Upgrade** to confirm.

Finalize the HDFS Upgrade

To determine if you can finalize the upgrade, run important workloads and ensure that they are successful. After you have finalized the upgrade, you cannot roll back to a previous version of HDFS without using backups. Verifying that you are ready to finalize the upgrade can take a long time.

Make sure you have enough free disk space, keeping in mind that the following behavior continues until the upgrade is finalized:

- Deleting files does not free up disk space.
- Using the balancer causes all moved replicas to be duplicated.
- All on-disk data representing the NameNodes metadata is retained, which could more than double the amount of space required on the NameNode and JournalNode disks.

If you have Enabled high availability for HDFS, and you have performed a rolling upgrade:

1. Go to the HDFS service.
2. Select **Actions** > **Finalize Rolling Upgrade** and click **Finalize Rolling Upgrade** to confirm.

If you have not performed a rolling upgrade:

1. Go to the HDFS service.
2. Click the **Instances** tab.
3. Click the link for the **NameNode** instance. If you have enabled high availability for HDFS, click the link labeled **NameNode (Active)**.

The NameNode instance page displays.

4. Select **Actions** > **Finalize Metadata Upgrade** and click **Finalize Metadata Upgrade** to confirm.

For Sentry with an Oracle Database, Add the AUTHZ_PATH.AUTHZ_OBJ_ID Index

If your cluster uses **Sentry** and an Oracle database, you must manually add the index on the AUTHZ_PATH.AUTHZ_OBJ_ID column if it does not already exist. Adding the index manually decreases the time Sentry takes to get a full snapshot for HDFS sync. Use the following command to add the index:

```
CREATE INDEX "AUTHZ_PATH_FK_IDX" ON "AUTHZ_PATH" ("AUTHZ_OBJ_ID");
```

Complete Post-Upgrade Migration Steps

Several components require additional migrations steps after you complete the CDH upgrade:

- **Impala** – See [Impala Upgrade Considerations](#) on page 106
- **Cloudera Search**

After upgrading to CDH 6, you must re-index your collections, see [Re-Indexing Solr Collections After Upgrading to CDH 6](#) on page 110.

- **Spark** – See [Apache Spark Post Upgrade Migration Steps](#) on page 110.
- **MapReduce 1 to MapReduce 2** – See [Migrating from MapReduce 1 \(MRv1\) to MapReduce 2 \(MRv2\)](#) on page 111
- **Kudu** – See [Upgrade Notes for Kudu 1.9 / CDH 6.2.0](#) on page 123
- **Kafka**

1. Remove the following properties from the **Kafka Broker Advanced Configuration Snippet (Safety Valve)** configuration property.

- `Inter.broker.protocol.version`
- `log.message.format.version`

2. Save your changes.

3. Restart the cluster:

- a. On the **Home > Status** tab, click




to the right of the cluster name and select **Restart**.

- b. Click **Restart** that appears in the next screen to confirm. If you have enabled high availability for HDFS, you can choose **Rolling Restart** instead to minimize cluster downtime. The **Command Details** window shows the progress of stopping services.

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

Exit Maintenance Mode

If you entered maintenance mode during this upgrade, exit maintenance mode.

On the **Home > Status** tab, click  next to the cluster name and select **Exit Maintenance Mode**.

CDH 6 Post-Upgrade Migration Steps

If you have deployed the Sentry, HBase, Cloudera Search, or Spark services in a CDH 5.x cluster that you want to upgrade to CDH 6, see the following topics for additional post-upgrade steps:

Cloudera Data Science Workbench: If you are using Cloudera Data Science Workbench, note that Cloudera Data Science Workbench does not automatically detect configuration changes on the CDH cluster. Perform a full reset of Cloudera Data Science Workbench so that it can pick up any changes as a result of the upgrade. For instructions, see the [associated known issue](#) in the Cloudera Data Science Workbench documentation.

Impala Upgrade Considerations

Converting Legacy UDFs During Upgrade to CDH 5.12 or Higher

In CDH 5.7 / Impala 2.5 and higher, the [CREATE FUNCTION Statement](#) is available for creating Java-based UDFs. UDFs created with the new syntax persist across Impala restarts, and are more compatible with Hive UDFs. Because the replication features in CDH 5.12 and higher only work with the new-style syntax, convert any older Java UDFs to use the new syntax at the same time you upgrade to CDH 5.12 or higher.

Follow these steps to convert old-style Java UDFs to the new persistent kind:

- Use `SHOW FUNCTIONS` to identify all UDFs and UDAs.
- For each function, use `SHOW CREATE FUNCTION` and save the statement in a script file.
- For Java UDFs, change the output of `SHOW CREATE FUNCTION` to use the new `CREATE FUNCTION` syntax (without argument types), which makes the UDF persistent.
- For each function, drop it and re-create it, using the new `CREATE FUNCTION` syntax for all Java UDFs.

Handling Large Rows During Upgrade to CDH 5.13 / Impala 2.10 or Higher

In CDH 5.13 / Impala 2.10 and higher, the handling of memory management for large column values is different than in previous releases. Some queries that succeeded previously might now fail immediately with an error message. The `--read_size` option no longer needs to be increased from its default of 8 MB for queries against tables with huge column values. Instead, the query option `MAX_ROW_SIZE` lets you fine-tune this value at the level of individual queries or sessions. The default for `MAX_ROW_SIZE` is 512 KB. If your queries process rows with column values totalling more than 512 KB, you might need to take action to avoid problems after upgrading.

Follow these steps to verify if your deployment needs any special setup to deal with the new way of dealing with large rows:

1. Check if your `impalad` daemons are already running with a larger-than-normal value for the `--read_size` configuration setting.
2. Examine all tables to find if any have `STRING` values that are hundreds of kilobytes or more in length. This information is available under the `Max Size` column in the output from the `SHOW TABLE STATS` statement, after the `COMPUTE STATS` statement has been run on the table. In the following example, the `s1` column with a maximum length of 700006 could cause an issue by itself, or if a combination of values from the `s1`, `s2`, and `s3` columns exceeded the 512 KB `MAX_ROW_SIZE` value.

```
show column stats big_strings;
```

Column	Type	#Distinct Values	#Nulls	Max Size	Avg Size
x	BIGINT	30000	-1	8	8
s1	STRING	30000	-1	700006	392625
s2	STRING	30000	-1	10532	9232.6669921875
s3	STRING	30000	-1	103	87.66670227050781

3. For each candidate table, run a query to materialize the largest string values from the largest columns all at once. Check if the query fails with a message suggesting to set the `MAX_ROW_SIZE` query option.

```
select count(distinct s1, s2, s3) from little_strings;
```

count(distinct s1, s2, s3)
30000

```
select count(distinct s1, s2, s3) from big_strings;
```

WARNINGS: Row of size 692.13 KB could not be materialized in plan node with id 1.
Increase the `max_row_size` query option (currently 512.00 KB) to process larger rows.

If any of your tables are affected, make sure the `MAX_ROW_SIZE` is set large enough to allow all queries against the affected tables to deal with the large column values:

- In SQL scripts run by `impala-shell` with the `-q` or `-f` options, or in interactive `impala-shell` sessions, issue a statement `SET MAX_ROW_SIZE=large_enough_size` before the relevant queries:

```
$ impala-shell -i localhost -q \  
'set max_row_size=1mb; select count(distinct s1, s2, s3) from big_strings'
```

- If large column values are common to many of your tables and it is not practical to set `MAX_ROW_SIZE` only for a limited number of queries or scripts, use the `--default_query_options` configuration setting for all your `impalad` daemons, and include the larger `MAX_ROW_SIZE` setting as part of the argument to that setting. For example:

```
impalad --default_query_options='max_row_size=1gb;appx_count_distinct=true'
```

- If your deployment uses a non-default value for the `--read_size` configuration setting, remove that setting and let Impala use the default. A high value for `--read_size` could cause higher memory consumption in CDH 5.13 / Impala 2.10 and higher than in previous versions. The `--read_size` setting still controls the HDFS I/O read size (which is rarely if ever necessary to change), but no longer affects the spill-to-disk buffer size.

Change Impala catalogd Heap when Upgrading from CDH 5.6 or Lower

The default heap size for Impala `catalogd` has changed in CDH 5.7 / Impala 2.5 and higher:

- Before 5.7, by default `catalogd` was using the JVM's default heap size, which is the smaller of 1/4th of the physical memory or 32 GB.
- Starting with CDH 5.7.0, the default `catalogd` heap size is 4 GB.

For example, on a host with 128 GB physical memory this will result in `catalogd` heap decreasing from 32 GB to 4 GB.

For schemas with large numbers of tables, partitions, and data files, the `catalogd` daemon might encounter an out-of-memory error. To prevent the error, increase the memory limit for the `catalogd` daemon:

1. Check current memory usage for the `catalogd` daemon by running the following commands on the host where that daemon runs on your cluster:

```
jcmd catalogd_pid VM.flags  
jmap -heap catalogd_pid
```

2. Decide on a large enough value for the `catalogd` heap.

- On systems managed by Cloudera Manager, include this value in the configuration field **Java Heap Size of Catalog Server in Bytes** (Cloudera Manager 5.7 and higher), or **Impala Catalog Server Environment Advanced Configuration Snippet (Safety Valve)** (prior to Cloudera Manager 5.7). Then restart the Impala service.
- On systems not managed by Cloudera Manager, put the `JAVA_TOOL_OPTIONS` environment variable setting into the startup script for the `catalogd` daemon, then restart the `catalogd` daemon.

For example, the following environment variable setting specifies the maximum heap size of 8 GB.

```
JAVA_TOOL_OPTIONS="-Xmx8g"
```

3. Use the same `jcmd` and `jmap` commands as earlier to verify that the new settings are in effect.

List of Reserved Words Updated in CDH 6.0 / Impala 3.0

The list of [Impala Reserved Words](#) in Impala was updated in CDH 6.0 / Impala 3.0. If you need to use a reserved word as an identifier, e.g. a table name, enclose the word in back-ticks.

If you need to use the reserved words from previous versions of CDH, set the `impalad` and `catalogd` startup option, `--reserved_words_version`, to "2.11.0".

Decimal V2 Used by Default in CDH 6.0 / Impala 3.0

In Impala, two different behaviors of `DECIMAL` types are supported. In CDH 6.0 / Impala 3.0, `DECIMAL V2` is used by default. See [DECIMAL Data Type](#) for detail information.

If you need to continue using the first version of the `DECIMAL` type for the backward compatibility of your queries, set the `DECIMAL_V2` query option to `FALSE`:

```
SET DECIMAL_V2=FALSE;
```

Behavior of Column Aliases Changed in CDH 6.0 / Impala 3.0

To conform to the SQL standard, Impala no longer performs alias substitution in the subexpressions of `GROUP BY`, `HAVING`, and `ORDER BY`. See [Overview of Impala Aliases](#) for examples of supported and unsupported aliases syntax.

Default PARQUET_ARRAY_RESOLUTION Changed in CDH 6.0 / Impala 3.0

The default value for the `PARQUET_ARRAY_RESOLUTION` was changed to `THREE_LEVEL` in CDH 6.0 / Impala 3.0, to match the Parquet standard 3-level encoding.

See [PARQUET_RESOLUTION Query option](#) for the information about the query option.

Enable Clustering Hint for Inserts

In CDH 6.0 / Impala 3.0, the [clustered](#) hint is enabled by default. The hint adds a local sort by the partitioning columns to a query plan.

The `clustered` hint is only effective for HDFS and Kudu tables.

As in previous versions, the `noclustered` hint prevents clustering. If a table has ordering columns defined, the `noclustered` hint is ignored with a warning.

Deprecated Query Options Removed in CDH 6.0 / Impala 3.0

The following query options have been deprecated for several releases and removed:

- `DEFAULT_ORDER_BY_LIMIT`
- `ABORT_ON_DEFAULT_LIMIT_EXCEEDED`
- `V_CPU_CORES`
- `RESERVATION_REQUEST_TIMEOUT`
- `RM_INITIAL_MEM`
- `SCAN_NODE_CODEGEN_THRESHOLD`
- `MAX_IO_BUFFERS`
- `RM_INITIAL_MEM`
- `DISABLE_CACHED_READS`

refresh_after_connect Impala Shell Option Removed in CDH 6.0 / Impala 3.0

The deprecated `refresh_after_connect` option was removed from Impala Shell in CDH 6.0 / Impala 3.0

Return Type Changed for EXTRACT and DATE_PART Functions in CDH 6.0 / Impala 3.0

The following changes were made to the `EXTRACT` and `DATE_PART` functions:

- The output type of the `EXTRACT` and `DATE_PART` functions was changed to `BIGINT`.
- Extracting the millisecond part from a `TIMESTAMP` returns the seconds component and the milliseconds component. For example, `EXTRACT (CAST('2006-05-12 18:27:28.123456789' AS TIMESTAMP), 'MILLISECOND')` will return 28123.

Impala Roles with SELECT or INSERT Privilege Receive REFRESH Privilege During the Upgrade to CDH 5.16 / CDH 6.1

Due to the Sentry and Impala fine grained privileges feature in CDH 5.16 / CDH 6.1, if a role has the `SELECT` or `INSERT` privilege on an object in Impala before upgrading to CDH 5.16 / CDH 6.1, that role will automatically get the `REFRESH` privilege during the upgrade.

Port Change for SHUTDOWN Command

If you used the `SHUTDOWN` command in CDH 6.1, and specified a port explicitly, change the port number parameter, in CDH 6.2, to use the KRPC port.

Default Setting Changes

Release Changed	Setting	Default Value
CDH 5.15 & CDH 6.1 / Impala 2.12	<code>--compact_catalog_topic</code> <code>impalad</code> flag	true
CDH 6.1 / Impala 2.12	<code>--max_cached_file_handles</code> <code>impalad</code> flag	20000
CDH 6.0 / Impala 3.0	<code>PARQUET_ARRAY_RESOLUTION</code> query option	THREE_LEVEL
	<code>DECIMAL_V2</code> query option	TRUE

Re-Indexing Solr Collections After Upgrading to CDH 6

After upgrading to CDH 6, your Solr collections will be automatically recreated with the updated configuration, but the collections are empty. Cloudera does not support upgrading the underlying index files from CDH 5 to CDH 6, so you must re-index your collections.

Apache Solr [does not have](#) dedicated re-indexing functionality. To re-index a collection, you must perform whatever indexing procedure you initially used to index your documents. For more information on indexing, see [Indexing Data Using Cloudera Search](#).

If you are storing documents in Cloudera Search itself, contact Cloudera Support for assistance.

Apache Spark Post Upgrade Migration Steps

After upgrading to CDH 6, you might have multiple Spark services configured, each with their own set of configurations, including event log locations. Decide which service to keep and then manually merge the two services.



Important:

- After the upgrade, both Spark services will use the same version of Spark.
- The command used to submit Spark 2 jobs in CDH 5 (`spark2-submit`) is removed in CDH 6, replaced by `spark-submit`. In a CDH 5 cluster with both the built-in Spark 1.6 service and a Spark 2 service, `spark-submit` is used with the Spark 1.6 service, and `spark2-submit` is used with the Spark 2 service. After upgrading to CDH 6, `spark-submit` uses the CDH built-in Spark 2 service, and `spark2-submit` does not work. Make sure to update any workflows that submit Spark jobs using these commands.
- After the upgrade, the `/etc/spark/conf` directory on gateway nodes includes the client configurations from the Spark service configured with highest alternatives priority before the upgrade. (See [Set Alternatives Priorities for Built-in CDH 5 Spark \(1.6\) and CDS 2](#) on page 86.)
- When you delete the service in Cloudera Manager, it does not delete the related event logs in HDFS.

Manually merge your Spark services by performing the following steps:

1. Copy all relevant configurations from the service you are removing to the service you are keeping. To view and edit the configurations:
 - a. In the Cloudera Manager Admin Console, go to the Spark service you are removing.

- b. Click the **Configuration** tab.
- c. Note the configurations.
- d. Go to the Spark service you are keeping and replicate the configuration.
- e. Click **Save Changes**.

2. To preserve historic event logs:

- a. Identify the location of the event logs for the service you are removing:
 - a. In the Cloudera Manager Admin Console, go to the Spark service you are removing.
 - b. Click the **Configuration** tab.
 - c. Search for: **spark.eventLog.dir**
 - d. Note the path.

- b. Log into a cluster host and run the following command:

```
hadoop fs -mv <old_Spark_Event_Log_dir>/* <new_location>/. 
```

3. Using Cloudera Manager, stop and delete the Spark Service you have selected for removal:

- a. In the Cloudera Manager Admin Console, click the drop-down arrow next to the Spark service you are removing, and then select **Stop**.
- b. Click the drop-down arrow next to the Spark service you are removing, and then select **Delete**.

4. Restart the remaining Spark service: Click the drop-down arrow next to the Spark service, and then select **Restart**.

Migrating from MapReduce 1 (MRv1) to MapReduce 2 (MRv2)



Note: This page contains references to CDH 5 components or features that have been removed from CDH 6. These references are only applicable if you are managing a CDH 5 cluster with Cloudera Manager 6. For more information, see [Deprecated Items](#).

CDH 6 does not support Apache Hadoop MapReduce version 1. This is a guide to migrate from MapReduce 1 (MRv1) to MapReduce 2 (MRv2) with Apache Hadoop YARN.

Introduction

MapReduce 2 is an upgrade to the way that scheduling, resource management, and execution occur in Hadoop. At their core, the improvements separate cluster resource management capabilities from MapReduce-specific logic. They enable Hadoop to share resources dynamically between MapReduce and other parallel processing frameworks, such as Impala, allow more sensible and finer-grained resource configuration for better cluster utilization, and permit it to scale to accommodate more and larger jobs.

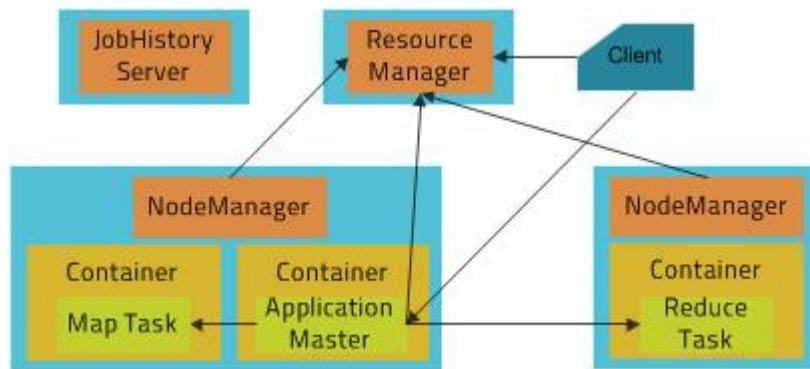
This document provides a guide to both the architectural and user-facing changes, so that both cluster operators and MapReduce programmers can easily make the transition.

Terminology and Architecture

MapReduce in Hadoop 2 was split into two components. The cluster resource management capabilities became YARN (Yet Another Resource Negotiator), while the MapReduce-specific capabilities remained MapReduce. In the MapReduce version 1 (MRv1) architecture, the cluster was managed by a service called the JobTracker. TaskTracker services lived on each host and would launch tasks on behalf of jobs. The JobTracker would serve information about completed jobs.

In MapReduce version 2 (MRv2), the functions of the JobTracker have been split between three services. The ResourceManager is a persistent YARN service that receives and runs applications (a MapReduce job is an application) on the cluster. It contains the scheduler, which, as previously, is pluggable. The MapReduce-specific capabilities of the JobTracker have been moved into the MapReduce ApplicationMaster, one of which is started to manage each MapReduce job and terminated when the job completes. The JobTracker function of serving information about completed jobs has been moved to the JobHistory Server. The TaskTracker has been replaced with the NodeManager, a YARN service

that manages resources and deployment on a host. The TaskTracker is responsible for launching containers, each of which can house a map or reduce task.



The new architecture has several advantages. For example, by breaking up the JobTracker into a few different services, it avoids many of the scaling issues faced by MRv1. More importantly, it makes it possible to run frameworks other than MapReduce on a Hadoop cluster. For example, Spark [can also](#) run on YARN.

For MapReduce Programmers: Writing and Running Jobs

Nearly all jobs written for MRv1 can run without any modifications on an MRv2 cluster.

CDH 6 supports applications compiled against MapReduce frameworks in CDH 5.7.0 and higher. Make sure that CDH jars are not included with your application by marking them as "provided" in the `pom.xml` file.

Java API Compatibility

MRv2 supports both the old (`mapred`) and new (`mapreduce`) MapReduce APIs used for MRv1, with a few caveats. The difference between the old and new APIs, which concerns user-facing changes, should not be confused with the difference between MRv1 and MRv2, which concerns changes to the underlying framework. CDH 5 supports the new and old MapReduce APIs.

In general, applications that use `@Public/@Stable` APIs are binary-compatible in CDH 5, meaning that compiled binaries should be able to run without modifications on the new framework. Source compatibility might be broken for applications that use a few obscure APIs that are technically public, but rarely needed and primarily exist for internal use. These APIs are detailed below. Source incompatibility means that code changes are required to compile. Source incompatibility is orthogonal to binary compatibility. Binaries for an application that is binary-compatible, but not source-compatible, continue to run on the new framework, but you must update your code to regenerate those binaries.

	Binary Incompatibilities	Source Incompatibilities
CDH 4 MRv1 to CDH 5 MRv1	None	None
CDH 4 MRv1 to CDH 5 MRv2	None	Rare
CDH 5 MRv1 to CDH 5 MRv2	None	Rare

The following are the known source incompatibilities:

- `KeyValueLineRecordReader#getProgress` and `LineRecordReader#getProgress` now throw `IOExceptions` in both the old and new APIs. Their superclass method, `RecordReader#getProgress`, already did this, but source compatibility will be broken for the rare code that used it without a `try/catch` block.
- `FileOutputCommitter#abortTask` now throws an `IOException`. Its superclass method always did this, but source compatibility will be broken for the rare code that used it without a `try/catch` block. This was fixed in CDH 4.3 MRv1 to be compatible with MRv2.
- `Job#getDependentJobs`, an API marked `@Evolving`, now returns a `List` instead of an `ArrayList`.

Compiling Jobs Against MRv2

If you are using Maven, compiling against MRv2 requires including the same artifact, `hadoop-client`. Changing the version to Hadoop 2 version (for example, using `2.2.0-cdh5.0.0` instead of `2.0.0-mr1-cdh4.3.0`) should be enough. If you are not using Maven, compiling against all the Hadoop JARs is recommended. A comprehensive list of Hadoop Maven artifacts is available. See [Maven Artifacts for CDH 6.0.x Releases](#).

If you want your job to run against both MRv1 and MRv2, compile it against MRv2.

Job Configuration

As in MRv1, job configuration options can be specified on the command line, in Java code, or in the `mapred-site.xml` on the client machine in the same way they previously were. The vast majority of job configuration options that were available in MRv1 work in MRv2 as well. For consistency and clarity, many options have been given new names. The older names are deprecated, but will still work for the time being. The exceptions to this are `mapred.child.ulimit` and all options relating to JVM reuse, as these are no longer supported.

Submitting and Monitoring Jobs

The MapReduce command line interface remains entirely compatible. Use of the Hadoop command line tool to run MapReduce related commands (`job`, `queue`, `classpath`, `historyserver`, `distcp`, `archive`) is deprecated, but still works. The `mapred` command line tool is preferred for these commands.

Selecting Appropriate JAR files for Your Jobs

The following table shows the names and locations of the JAR files used in MRv1 and the corresponding names and locations in YARN:

Name	MapReduce MRv1 location	YARN location
Streaming	<code>/usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh<version>.jar</code>	<code>/usr/lib/hadoop-mapreduce/hadoop-streaming.jar</code>
Rumen	N/A	<code>/usr/lib/hadoop-mapreduce/hadoop-rumen.jar</code>
Hadoop Examples	<code>/usr/lib/hadoop-0.20-mapreduce/hadoop-examples.jar</code>	<code>/usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar</code>
DistCp v1	<code>/usr/lib/hadoop-0.20-mapreduce/hadoop-tools.jar</code>	<code>/usr/lib/hadoop-mapreduce/hadoop-extras.jar</code>
DistCp v2	N/A	<code>/usr/lib/hadoop-mapreduce/hadoop-distcp.jar</code>
Hadoop archives	<code>/usr/lib/hadoop-0.20-mapreduce/hadoop-tools.jar</code>	<code>/usr/lib/hadoop-mapreduce/hadoop-archives.jar</code>

Requesting Resources

A MapReduce job submission includes the amount of resources to reserve for each map and reduce task. As in MapReduce 1, the amount of memory requested is controlled by the `mapreduce.map.memory.mb` and `mapreduce.reduce.memory.mb` properties.

MapReduce 2 adds additional parameters that control how much processing power to reserve for each task as well. The `mapreduce.map.cpu.vcores` and `mapreduce.reduce.cpu.vcores` properties express how much parallelism a map or reduce task can take advantage of. These should remain at their default value of 1 unless your code is explicitly spawning extra compute-intensive threads.

For Administrators: Configuring and Running MRv2 Clusters Configuration Migration

Since MapReduce 1 functionality has been split into two components, MapReduce cluster configuration options have been split into YARN configuration options, which go in `yarn-site.xml`, and MapReduce configuration options, which go in `mapred-site.xml`. Many have been given new names to reflect the shift. As JobTrackers and TaskTrackers no

longer exist in MRv2, all configuration options pertaining to them no longer exist, although many have corresponding options for the ResourceManager, NodeManager, and JobHistoryServer.

A minimal configuration required to run MRv2 jobs on YARN is:

- yarn-site.xml configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>you.hostname.com</value>
  </property>

  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

- mapred-site.xml configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

Resource Configuration

One of the larger changes in MRv2 is the way that resources are managed. In MRv1, each host was configured with a fixed number of map slots and a fixed number of reduce slots. Under YARN, there is no distinction between resources available for maps and resources available for reduces - all resources are available for both. Second, the notion of slots has been discarded, and resources are now configured in terms of amounts of memory (in megabytes) and CPU (in “virtual cores”, which are described below). Resource configuration is an inherently difficult topic, and the added flexibility that YARN provides in this regard also comes with added complexity. Cloudera Manager will pick sensible values automatically, but if you are setting up your cluster manually or just interested in the details, read on.

Configuring Memory Settings for YARN and MRv2

The memory configuration for YARN and MRv2 memory is important to get the best performance from your cluster. Several different settings are involved. The table below shows the default settings, as well as the settings that Cloudera recommends, for each configuration option. See [Managing YARN \(MRv2\) and MapReduce \(MRv1\)](#) for more configuration specifics; and, for detailed tuning advice with sample configurations, see [Tuning YARN](#).

Table 2: YARN and MRv2 Memory Configuration

Cloudera Manager Property Name	CDH Property Name	Default Configuration	Cloudera Tuning Guidelines
Container Memory Minimum	yarn.scheduler.minimum-allocation-mb	1 GB	0
Container Memory Maximum	yarn.scheduler.maximum-allocation-mb	64 GB	amount of memory on largest host
Container Memory Increment	yarn.scheduler.increment-allocation-mb	512 MB	Use a fairly large value, such as 128 MB
Container Memory	yarn.nodemanager.resource.memory-mb	8 GB	8 GB
Map Task Memory	mapreduce.map.memory.mb	1 GB	1 GB

Cloudera Manager Property Name	CDH Property Name	Default Configuration	Cloudera Tuning Guidelines
Reduce Task Memory	<code>mapreduce.reduce.memory.mb</code>	1 GB	1 GB
Map Task Java Opts Base	<code>mapreduce.map.java.opts</code>	<code>-Djava.net.preferIPv4Stack=true</code>	<code>-Djava.net.preferIPv4Stack=true</code> <code>-Xmx768m</code>
Reduce Task Java Opts Base	<code>mapreduce.reduce.java.opts</code>	<code>-Djava.net.preferIPv4Stack=true</code>	<code>-Djava.net.preferIPv4Stack=true</code> <code>-Xmx768m</code>
ApplicationMaster Memory	<code>yarn.app.mapreduce.am.resource.mb</code>	1 GB	1 GB
ApplicationMaster Java Opts Base	<code>yarn.app.mapreduce.am.command-opts</code>	<code>-Djava.net.preferIPv4Stack=true</code>	<code>-Djava.net.preferIPv4Stack=true</code> <code>-Xmx768m</code>

Resource Requests

From the perspective of a developer requesting resource allocations for a job's tasks, nothing needs to be changed. Map and reduce task memory requests still work and, additionally, tasks that will use multiple threads can request more than 1 core with the `mapreduce.map.cpu.vcores` and `mapreduce.reduce.cpu.vcores` properties.

Configuring Host Capacities

In MRv1, the `mapred.tasktracker.map.tasks.maximum` and `mapred.tasktracker.reduce.tasks.maximum` properties dictated how many map and reduce slots each TaskTracker had. These properties no longer exist in YARN. Instead, YARN uses `yarn.nodemanager.resource.memory-mb` and `yarn.nodemanager.resource.cpu-vcores`, which control the amount of memory and CPU on each host, both available to both maps and reduces. If you were using Cloudera Manager to configure these automatically, Cloudera Manager will take care of it in MRv2 as well. If configuring these manually, simply set these to the amount of memory and number of cores on the machine after subtracting out resources needed for other services.

Virtual Cores

To better handle varying CPU requests, YARN supports virtual cores (vcores), a resource meant to express parallelism. The "virtual" in the name is somewhat misleading - on the NodeManager, vcores should be configured equal to the number of physical cores on the machine. Tasks should be requested with vcores equal to the number of cores they can saturate at once. Currently vcores are very coarse - tasks will rarely want to ask for more than one of them, but a complementary axis that represents processing power may be added in the future to enable finer-grained resource configuration.

Rounding Request Sizes

Also noteworthy are the `yarn.scheduler.minimum-allocation-mb`, `yarn.scheduler.minimum-allocation-vcores`, `yarn.scheduler.increment-allocation-mb`, and `yarn.scheduler.increment-allocation-vcores` properties, which default to 1024, 1, 512, and 1 respectively. If tasks are submitted with resource requests lower than the minimum-allocation values, their requests will be set to these values. If tasks are submitted with resource requests that are not multiples of the increment-allocation values, their requests will be rounded up to the nearest increments.

To make all of this more concrete, let's use an example. Each host in the cluster has 24 GB of memory and 6 cores. Other services running on the hosts require 4 GB and 1 core, so we set `yarn.nodemanager.resource.memory-mb` to 20480 and `yarn.nodemanager.resource.cpu-vcores` to 5. If you leave the map and reduce task defaults of 1024 MB and 1 virtual core intact, you will have at most 5 tasks running at the same time. If you want each of your tasks to use 5 GB, set their `mapreduce.(map|reduce).memory.mb` to 5120, which would limit you to 4 tasks running at the same time.

Scheduler Configuration

Cloudera recommends using the Fair Scheduler in MRv2. (FIFO and Capacity Scheduler are also available.) Fair Scheduler allocation files require changes in light of the new way that resources work. The `minMaps`, `maxMaps`, `minReduces`, and `maxReduces` queue properties have been replaced with a `minResources` property and a `maxProperties`. Instead of taking a number of slots, these properties take a value like “1024 MB, 3 vcores”. By default, the MRv2 Fair Scheduler will attempt to equalize memory allocations in the same way it attempted to equalize slot allocations in MRv1. The MRv2 Fair Scheduler contains a number of new features including hierarchical queues and fairness based on multiple resources.

For more information on tuning and resource management, see [Tuning YARN](#) and [YARN \(MRv2\) and MapReduce \(MRv1\) Schedulers](#).

Administration Commands

The `jobtracker` and `tasktracker` commands, which start the JobTracker and TaskTracker, are no longer supported because these services no longer exist. They are replaced with `yarn resourcemanager` and `yarn nodemanager`, which start the ResourceManager and NodeManager respectively. `hadoop mradmin` is no longer supported. Instead, `yarn rmadmin` should be used. The new admin commands mimic the functionality of the MRv1 names, allowing hosts, queues, and ACLs to be refreshed while the ResourceManager is running.

Security

The following section outlines the additional changes needed to migrate a secure cluster.

New YARN Kerberos service principals should be created for the ResourceManager and NodeManager, using the pattern used for other Hadoop services, that is, `yarn@HOST`. The `mapred` principal should still be used for the JobHistory Server. If you are using Cloudera Manager to configure security, this will be taken care of automatically.

As in MRv1, a configuration must be set to have the user that submits a job own its task processes. The equivalent of the MRv1 `LinuxTaskController` is the `LinuxContainerExecutor`. In a secure setup, NodeManager configurations should set `yarn.nodemanager.container-executor.class` to `org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor`. Properties set in the `taskcontroller.cfg` configuration file should be migrated to their analogous properties in the `container-executor.cfg` file.

In secure setups, configuring `hadoop-policy.xml` allows administrators to set up access control lists on internal protocols. The following is a table of MRv1 options and their MRv2 equivalents:

MRv1	MRv2	Comment
<code>security.task.umbilical.protocol.acl</code>	<code>security.job.task.protocol.acl</code>	As in MRv1, this should never be set to anything other than *
<code>security.inter.tracker.protocol.acl</code>	<code>security.resourcetracker.protocol.acl</code>	
<code>security.job.submission.protocol.acl</code>	<code>security.applicationclient.protocol.acl</code>	
<code>security.admin.operations.protocol.acl</code>	<code>security.resourcemanager-administration.protocol.acl</code>	
	<code>security.applicationmaster.protocol.acl</code>	No MRv1 equivalent
	<code>security.containermanagement.protocol.acl</code>	No MRv1 equivalent
	<code>security.resourcelocalizer.protocol.acl</code>	No MRv1 equivalent
	<code>security.job.client.protocol.acl</code>	No MRv1 equivalent

Queue access control lists (ACLs) are now placed in the Fair Scheduler configuration file instead of the JobTracker configuration. A list of users and groups that can submit jobs to a queue can be placed in `aclSubmitApps` in the queue’s configuration. The queue administration ACL is no longer supported, but will be in a future release.

Ports

The following is a list of default ports used by MRv2 and YARN, as well as the configuration properties used to configure them.

Port	Use	Property
8032	ResourceManager Client RPC	yarn.resourcemanager.address
8030	ResourceManager Scheduler RPC (for ApplicationMasters)	yarn.resourcemanager.scheduler.address
8033	ResourceManager Admin RPC	yarn.resourcemanager.admin.address
8088	ResourceManager Web UI and REST APIs	yarn.resourcemanager.webapp.address
8031	ResourceManager Resource Tracker RPC (for NodeManagers)	yarn.resourcemanager.resource-tracker.address
8040	NodeManager Localizer RPC	yarn.nodemanager.localizer.address
8042	NodeManager Web UI and REST APIs	yarn.nodemanager.webapp.address
10020	Job History RPC	mapreduce.jobhistory.address
19888	Job History Web UI and REST APIs	mapreduce.jobhistory.webapp.address
13562	Shuffle HTTP	mapreduce.shuffle.port



Note: You can set `yarn.resourcemanager.hostname.id` for each ResourceManager instead of setting the ResourceManager values; this will cause YARN to use the default ports on those hosts.

High Availability

YARN supports ResourceManager HA to make a YARN cluster highly-available; the underlying architecture of active-standby pair is similar to JobTracker HA in MRv1. A major improvement over MRv1 is: in YARN, the completed tasks of in-flight MapReduce jobs are not re-run on recovery after the ResourceManager is restarted or failed over. Further, the configuration and setup has also been simplified. The main differences are:

1. Failover controller has been moved from a separate ZKFC daemon to be a part of the ResourceManager itself. So, there is no need to run an additional daemon.
2. Clients, applications, and NodeManagers do not require configuring a proxy-provider to talk to the active ResourceManager.

Below is a table with HA-related configurations used in MRv1 and their equivalents in YARN:

MRv1	YARN / MRv2	Comment
mapred.jobtrackers.name	yarn.resourcemanager.ha.rm-ids	
mapred.ha.jobtracker.id	yarn.resourcemanager.ha.id	Unlike in MRv1, this must be configured in YARN.
mapred.jobtracker.rpc-address.name.id	(See YARN (MRv2) ResourceManager High Availability)	YARN/ MRv2 has different RPC ports for different functionalities. Each port-related configuration must be suffixed with an id.

MRv1	YARN / MRv2	Comment
		There is no <i>name</i> component in YARN.
mapred.ha.jobtracker.rpc-address.name.id	yarn.resourcemanager.ha.admin.address	
mapred.ha.fencing.methods	yarn.resourcemanager.ha.fencer	Not required to be specified
mapred.client.failover.*	None	Not required
	yarn.resourcemanager.ha.enabled	Enable HA
mapred.jobtracker.restart.recover	yarn.resourcemanager.recovery.enabled	Enable recovery of jobs after failover
	yarn.resourcemanager.store.class	org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore
mapred.ha.automatic-failover.enabled	yarn.resourcemanager.ha.automatic-failover.enabled	Enable automatic failover
mapred.ha.zkfc.port	yarn.resourcemanager.ha.automatic-failover.port	
mapred.job.tracker	yarn.resourcemanager.cluster.id	Cluster name

Upgrading an MRv1 Installation Using Cloudera Manager

See [Importing MapReduce Configurations to YARN](#) for instructions.

Upgrading an MRv1 Installation Using the Command Line

1. Uninstall the following packages: `hadoop-0.20-mapreduce`, `hadoop-0.20-mapreduce-jobtracker`, `hadoop-0.20-mapreduce-tasktracker`, `hadoop-0.20-mapreduce-zkfc`, `hadoop-0.20-mapreduce-jobtrackerha`.
2. Install the following additional packages : `hadoop-yarn`, `hadoop-mapreduce`, `hadoop-mapreduce-historyserver`, `hadoop-yarn-resourcemanager`, `hadoop-yarn-nodemanager`.
3. Look at all the service configurations placed in `mapred-site.xml` and replace them with their corresponding YARN configuration. Configurations starting with `yarn` should be placed inside `yarn-site.xml`, not `mapred-site.xml`. Refer to [Resource Configuration](#) for best practices on how to convert TaskTracker slot capacities (`mapred.tasktracker.map.tasks.maximum` and `mapred.tasktracker.reduce.tasks.maximum`) to NodeManager resource capacities (`yarn.nodemanager.resource.memory-mb` and `yarn.nodemanager.resource.cpu-vcores`), as well as how to convert configurations in the Fair Scheduler allocations file, `fair-scheduler.xml`.
4. Start the ResourceManager, NodeManagers, and the JobHistoryServer.

Web UI

In MRv1, the JobTracker Web UI served detailed information about the state of the cluster and the jobs (recent and current) running on it. It also contained the job history page, which served information from disk about older jobs.

The MRv2 Web UI provides the same information structured in the same way, but has been revamped with a new look and feel. The ResourceManager's UI, which includes information about running applications and the state of the cluster, is now located by default at `<ResourceManager host>:8088`. The JobHistory UI is now located by default at `<JobHistoryServer host>:19888`. Jobs can be searched and viewed there just as they could in MRv1.

Because the ResourceManager is meant to be agnostic to many of the concepts in MapReduce, it cannot host job information directly. Instead, it proxies to a Web UI that can. If the job is running, this proxy is the relevant MapReduce ApplicationMaster; if the job has completed, then this proxy is the JobHistoryServer. Thus, the user experience is similar to that of MRv1, but the information is now coming from different places.

Summary of Configuration Changes

The following tables summarize the changes in configuration parameters between MRv1 and MRv2.

JobTracker Properties and ResourceManager Equivalents

MRv1	YARN / MRv2
mapred.jobtracker.taskScheduler	yarn.resourcemanager.scheduler.class
mapred.jobtracker.completeuserjobs.maximum	yarn.resourcemanager.max-completed-applications
mapred.jobtracker.restart.recover	yarn.resourcemanager.recovery.enabled
mapred.job.tracker	yarn.resourcemanager.hostname or all of the following: yarn.resourcemanager.address yarn.resourcemanager.scheduler.address yarn.resourcemanager.resource-tracker.address yarn.resourcemanager.admin.address
mapred.job.tracker.http.address	yarn.resourcemanager.webapp.address or yarn.resourcemanager.hostname
mapred.job.tracker.handler.count	yarn.resourcemanager.resource-tracker.client.thread-count
mapred.hosts	yarn.resourcemanager.nodes.include-path
mapred.hosts.exclude	yarn.resourcemanager.nodes.exclude-path
mapred.cluster.max.map.memory.mb	yarn.scheduler.maximum-allocation-mb
mapred.cluster.max.reduce.memory.mb	yarn.scheduler.maximum-allocation-mb
mapred.acls.enabled	yarn.acl.enable
mapreduce.cluster.acls.enabled	yarn.acl.enable

JobTracker Properties and JobHistoryServer Equivalents

MRv1	YARN / MRv2	Comment
mapred.job.tracker.retiredjobs.cache.size	mapreduce.jobhistory.joblist.cache.size	
mapred.job.tracker.jobhistory.lru.cache.size	mapreduce.jobhistory.loadedjobs.cache.size	
mapred.job.tracker.history.completed.location	mapreduce.jobhistory.done-dir	Local FS in MR1; stored in HDFS in MR2
hadoop.job.history.user.location	mapreduce.jobhistory.done-dir	
hadoop.job.history.location	mapreduce.jobhistory.done-dir	

JobTracker Properties and MapReduce ApplicationMaster Equivalents

MRv1	YARN / MRv2	Comment
mapreduce.jobtracker.staging.root.dir	yarn.app.mapreduce.am.staging-dir	Now configurable per job

TaskTracker Properties and NodeManager Equivalents

MRv1	YARN / MRv2
mapred.tasktracker.map.tasks.maximum	yarn.nodemanager.resource.memory-mb and

MRv1	YARN / MRv2
	yarn.nodemanager.resource.cpu-vcores
mapred.tasktracker.reduce.tasks.maximum	yarn.nodemanager.resource.memory-mb and yarn.nodemanager.resource.cpu-vcores
mapred.tasktracker.expiry.interval	yarn.nm.liveliness-monitor.expiry-interval-ms
mapred.tasktracker.resourcecalculatorplugin	yarn.nodemanager.container-monitor.resource-calculator.class
mapred.tasktracker.taskmemorymanager.monitoring-interval	yarn.nodemanager.container-monitor.interval-ms
mapred.tasktracker.tasks.sleep-time-before-sigkill	yarn.nodemanager.sleep-delay-before-sigkill.ms
mapred.task.tracker.task-controller	yarn.nodemanager.container-executor.class
mapred.local.dir	yarn.nodemanager.local-dirs
mapreduce.cluster.local.dir	yarn.nodemanager.local-dirs
mapred.disk.healthChecker.interval	yarn.nodemanager.disk-health-checker.interval-ms
mapred.healthChecker.script.path	yarn.nodemanager.health-checker.script.path
mapred.healthChecker.interval	yarn.nodemanager.health-checker.interval-ms
mapred.healthChecker.script.timeout	yarn.nodemanager.health-checker.script.timeout-ms
mapred.healthChecker.script.args	yarn.nodemanager.health-checker.script.opts
local.cache.size	yarn.nodemanager.localizer.cache.target-size-mb
mapreduce.tasktracker.cache.local.size	yarn.nodemanager.localizer.cache.target-size-mb

TaskTracker Properties and Shuffle Service Equivalents

The table that follows shows TaskTracker properties and their equivalents in the auxiliary shuffle service that runs inside NodeManagers.

MRv1	YARN / MRv2
tasktracker.http.threads	mapreduce.shuffle.max.threads
mapred.task.tracker.http.address	mapreduce.shuffle.port
mapred.tasktracker.indexcache.mb	mapred.tasktracker.indexcache.mb

Per-Job Configuration Properties

Many of these properties have new names in MRv2, but the MRv1 names will work for all properties except `mapred.job.restart.recover`.

MRv1	YARN / MRv2	Comment
io.sort.mb	mapreduce.task.io.sort.mb	MRv1 name still works
io.sort.factor	mapreduce.task.io.sort.factor	MRv1 name still works
io.sort.spill.percent	mapreduce.task.io.sort.spill.percent	MRv1 name still works
mapred.map.tasks	mapreduce.job.maps	MRv1 name still works
mapred.reduce.tasks	mapreduce.job.reduces	MRv1 name still works
mapred.job.map.memory.mb	mapreduce.map.memory.mb	MRv1 name still works
mapred.job.reduce.memory.mb	mapreduce.reduce.memory.mb	MRv1 name still works

MRv1	YARN / MRv2	Comment
mapred.map.child.log.level	mapreduce.map.log.level	MRv1 name still works
mapred.reduce.child.log.level	mapreduce.reduce.log.level	MRv1 name still works
mapred.inmem.merge.threshold	mapreduce.reduce.shuffle.merge.inmem.threshold	MRv1 name still works
mapred.job.shuffle.merge.percent	mapreduce.reduce.shuffle.merge.percent	MRv1 name still works
mapred.job.shuffle.input.buffer.percent	mapreduce.reduce.shuffle.input.buffer.percent	MRv1 name still works
mapred.job.reduce.input.buffer.percent	mapreduce.reduce.input.buffer.percent	MRv1 name still works
mapred.map.tasks.speculative.execution	mapreduce.map.speculative	Old one still works
mapred.reduce.tasks.speculative.execution	mapreduce.reduce.speculative	MRv1 name still works
mapred.min.split.size	mapreduce.input.fileinputformat.split.minsize	MRv1 name still works
keep.failed.task.files	mapreduce.task.files.preserve.failedtasks	MRv1 name still works
mapred.output.compress	mapreduce.output.fileoutputformat.compress	MRv1 name still works
mapred.map.output.compression.codec	mapreduce.map.output.compress.codec	MRv1 name still works
mapred.compress.map.output	mapreduce.map.output.compress	MRv1 name still works
mapred.output.compression.type	mapreduce.output.fileoutputformat.compress.type	MRv1 name still works
mapred.userlog.limit.kb	mapreduce.task.userlog.limit.kb	MRv1 name still works
jobclient.output.filter	mapreduce.client.output.filter	MRv1 name still works
jobclient.completion.poll.interval	mapreduce.client.completion.pollinterval	MRv1 name still works
jobclient.progress.monitor.poll.interval	mapreduce.client.progressmonitor.pollinterval	MRv1 name still works
mapred.task.profile	mapreduce.task.profile	MRv1 name still works
mapred.task.profile.maps	mapreduce.task.profile.maps	MRv1 name still works
mapred.task.profile.reduces	mapreduce.task.profile.reduces	MRv1 name still works
mapred.line.input.format.linespermap	mapreduce.input.lineinputformat.linespermap	MRv1 name still works
mapred.skip.attempts.to.start.skipping	mapreduce.task.skip.start.attempts	MRv1 name still works
mapred.skip.map.auto.incr.proc.count	mapreduce.map.skip.proc.count.autoincr	MRv1 name still works
mapred.skip.reduce.auto.incr.proc.count	mapreduce.reduce.skip.proc.count.autoincr	MRv1 name still works
mapred.skip.out.dir	mapreduce.job.skip.outdir	MRv1 name still works
mapred.skip.map.max.skip.records	mapreduce.map.skip.maxrecords	MRv1 name still works
mapred.skip.reduce.max.skip.groups	mapreduce.reduce.skip.maxgroups	MRv1 name still works
job.end.retry.attempts	mapreduce.job.end-notification.retry.attempts	MRv1 name still works
job.end.retry.interval	mapreduce.job.end-notification.retry.interval	MRv1 name still works
job.end.notification.url	mapreduce.job.end-notification.url	MRv1 name still works
mapred.merge.recordsBeforeProgress	mapreduce.task.merge.progress.records	MRv1 name still works
mapred.job.queue.name	mapreduce.job.queue.name	MRv1 name still works
mapred.reduce.slowstart.completed.maps	mapreduce.job.reduce.slowstart.completedmaps	MRv1 name still works
mapred.map.max.attempts	mapreduce.map.maxattempts	MRv1 name still works

MRv1	YARN / MRv2	Comment
mapred.reduce.max.attempts	mapreduce.reduce.maxattempts	MRv1 name still works
mapred.reduce.parallel.copies	mapreduce.reduce.shuffle.parallelcopies	MRv1 name still works
mapred.task.timeout	mapreduce.task.timeout	MRv1 name still works
mapred.max.tracker.failures	mapreduce.job.maxtaskfailures.per.tracker	MRv1 name still works
mapred.job.restart.recover	mapreduce.am.max-attempts	
mapred.combine.recordsBeforeProgress	mapreduce.task.combine.progress.records	MRv1 name should still work - see MAPREDUCE-5130

Miscellaneous Properties

MRv1	YARN / MRv2
mapred.heartbeats.in.second	yarn.resourcemanager.nodemangers.heartbeat-interval-ms
mapred.userlog.retain.hours	yarn.log-aggregation.retain-seconds

MRv1 Properties that have no MRv2 Equivalents

MRv1	Comment
mapreduce.tasktracker.group	
mapred.child.ulimit	
mapred.tasktracker.dns.interface	
mapred.tasktracker.dns.nameserver	
mapred.tasktracker.instrumentation	NodeManager does not accept instrumentation
mapred.job.reuse.jvm.num.tasks	JVM reuse no longer supported
mapreduce.job.jvm.numtasks	JVM reuse no longer supported
mapred.task.tracker.report.address	No need for this, as containers do not use IPC with NodeManagers, and ApplicationMaster ports are chosen at runtime
mapreduce.task.tmp.dir	No longer configurable. Now always <code>tmp/</code> (under container's local dir)
mapred.child.tmp	No longer configurable. Now always <code>tmp/</code> (under container's local dir)
mapred.temp.dir	
mapred.jobtracker.instrumentation	ResourceManager does not accept instrumentation
mapred.jobtracker.plugins	ResourceManager does not accept plugins
mapred.task.cache.level	
mapred.queue.names	These go in the scheduler-specific configuration files
mapred.system.dir	
mapreduce.tasktracker.cache.local.numberdirectories	
mapreduce.reduce.input.limit	
io.sort.record.percent	Tuned automatically (MAPREDUCE-64)

MRv1	Comment
mapred.cluster.map.memory.mb	Not necessary; MRv2 uses resources instead of slots
mapred.cluster.reduce.memory.mb	Not necessary; MRv2 uses resources instead of slots
mapred.max.tracker.blacklists	
mapred.jobtracker.maxtasks.per.job	Related configurations go in scheduler-specific configuration files
mapred.jobtracker.taskScheduler.maxRunningTasksPerJob	Related configurations go in scheduler-specific configuration files
io.map.index.skip	
mapred.user.jobconf.limit	
mapred.local.dir.minspacestart	
mapred.local.dir.minspacekill	
hadoop.rpc.socket.factory.class.JobSubmissionProtocol	
mapreduce.tasktracker.outofband.heartbeat	Always on
mapred.jobtracker.job.history.block.size	

Upgrade Notes for Kudu 1.9 / CDH 6.2.0



Important: Before upgrading, you should review the release notes ([CDH 5 release notes](#) or [CDH 6 release notes](#)) and the [platform requirements](#) for the version of Kudu that you are about to install.

When upgrading Kudu, it is recommended to first shut down all Kudu processes across the cluster, then upgrade the software on all servers, then restart the Kudu processes on all servers in the cluster.

- To improve security, world-readable Kerberos keytab files are no longer accepted by default. Set `--allow_world_readable_credentials=true` to override this behavior.

Wire Protocol compatibility

Kudu 1.9.0 is wire-compatible with previous versions of Kudu:

- Kudu 1.9 clients may connect to servers running Kudu 1.0 or later. If the client uses features that are not available on the target server, an error will be returned.
- Rolling upgrade between Kudu 1.8 and Kudu 1.9 servers is believed to be possible though has not been sufficiently tested. Users are encouraged to shut down all nodes in the cluster, upgrade the software, and then restart the daemons on the new version.
- Kudu 1.0 clients may connect to servers running Kudu 1.9 with the exception of the below-mentioned restrictions regarding secure clusters.

The authentication features introduced in Kudu 1.3 place the following limitations on wire compatibility between Kudu 1.9 and versions earlier than 1.3:

- If a Kudu 1.9 cluster is configured with authentication or encryption set to "required", clients older than Kudu 1.3 will be unable to connect.
- If a Kudu 1.9 cluster is configured with authentication and encryption set to "optional" or "disabled", older clients will still be able to connect.

Client Library Compatibility

- The Kudu 1.9 Java client library is API- and ABI-compatible with Kudu 1.8. Applications written against Kudu 1.8 will compile and run against the Kudu 1.9 client library and vice-versa.
- The Kudu 1.9 C++ client is API- and ABI-forward-compatible with Kudu 1.8. Applications written and compiled against the Kudu 1.8 client library will run without modification against the Kudu 1.9 client library. Applications written and compiled against the Kudu 1.9 client library will run without modification against the Kudu 1.8 client library.
- The Kudu 1.9 Python client is API-compatible with Kudu 1.8. Applications written against Kudu 1.8 will continue to run against the Kudu 1.9 client and vice-versa.

Location Awareness

Upon upgrading to CDH 6.2 / Kudu 1.9 or higher, if rack locations are assigned, you should run the `kudu cluster rebalance` tool to ensure your existing tables are in compliance with the rack awareness placement policy.

Upgrading CDH Manually after an Upgrade Failure



Important:

Perform the steps in this section only if the upgrade wizard reports a failure, or if you selected **Manual Upgrade** from the CDH Upgrade Wizard. Manual upgrades allow you to selectively stop and restart services to prevent or mitigate downtime for services or clusters where rolling restarts are not available.

All steps below assume the starting CDH version is at least 5.7.0, since that is the lowest CDH version that CM 6.0.x supports.

The steps below should be executed roughly in the order that they are listed, and should only be executed if the service is configured.

Start ZooKeeper

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the ZooKeeper service.
 2. Select **Actions** > **Start**.

Start Kudu

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the KUDU service.
 2. Select **Actions** > **Start**.

Upgrade HDFS Metadata

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the HDFS service.
 2. Select **Actions** > **Upgrade HDFS Metadata** and click **Upgrade HDFS Metadata** to confirm.

Start HDFS

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the HDFS service.
 2. Select **Actions** > **Start**.

Start HBASE

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the HBASE service.
 2. Select **Actions** > **Start**.

Upgrade the Sentry Database

Required for the following upgrades:

- CDH 5.x to 5.8.x
 - CDH 5.x to 5.9.x
 - CDH 5.x to 5.10.x
 - CDH 5.x to 5.13.x
 - CDH 5.x to 6.0.0 or greater
1. Go to the Sentry service.
 2. If the Sentry service is running, stop it:
 - a. Select **Actions** > **Stop** and click **Stop** to confirm.
 3. Select **Actions** > **Upgrade Sentry Database Tables** and click **Upgrade Sentry Database Tables** to confirm.
 4. If you are upgrading with an Oracle database you must manually add the index on the AUTHZ_PATH.AUTHZ_OBJ_ID column if it does not already exist. Adding the index manually decreases the time Sentry takes to get a full snapshot for HDFS sync. Use the following command to add the index:

```
CREATE INDEX "AUTHZ_PATH_FK_IDX" ON "AUTHZ_PATH" ( "AUTHZ_OBJ_ID" );
```

Start Sentry

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the Sentry service.
 2. Select **Actions** > **Start**.

Start KAFKA

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the KAFKA service.
 2. Select **Actions** > **Start**.

Upgrade Solr

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher

Upgrading CDH

1. If you have Sentry service in the cluster:
 - a. If the Solr service is running, stop it:
 - a. Select **Actions** > **Stop** and click **Stop** to confirm.
 - b. If the Sentry service is not running, start it:
 - a. Select **Actions** > **Start** and click **Start** to confirm.
 - c. Select **Actions** > **Migrate Sentry Privileges for Solr** and click **Migrate Sentry Privileges for Solr** to confirm
2. If the Sentry service is running, stop it:
 - a. Select **Actions** > **Stop** and click **Stop** to confirm.
3. If the Zookeeper service is not running, start it:
 - a. Select **Actions** > **Start** and click **Start** to confirm.
4. If the HDFS service is not running, start it:
 - a. Select **Actions** > **Start** and click **Start** to confirm.
5. Select **Actions** > **Reinitialize Solr State for Upgrade** and click **Reinitialize Solr State for Upgrade** to confirm.
6. Set the Solr configuration parameter **Solr Server for Upgrade** to the Solr service being upgraded
7. Select **Actions** > **Bootstrap Solr Configuration** and click **Bootstrap Solr Configuration** to confirm.

Start FLUME

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the FLUME service.
 2. Select **Actions** > **Start**.

Upgrade KeyStore Indexer

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the KeyStore Indexer service.
 2. Select **Actions** > **Migrate Policy file to Sentry** if enabled, and click **Migrate Policy file to Sentry** to confirm.

Start Key-Value Store Indexer

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the Key-Value Store Indexer service.
 2. Select **Actions** > **Start**.

Upgrade YARN

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Ensure that the ZooKeeper and HDFS services are running.
 2. Ensure that the YARN service is stopped.
 3. Go to the YARN service.
 4. Select **Actions** > **Clean NodeManager Recovery Directory** and click **Clean NodeManager Recovery Directory** to confirm.

Install MR Framework Jars

Required for all CDH upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the YARN service.
 2. Select **Actions** > **Install YARN MapReduce Framework JARs** and click **Install YARN MapReduce Framework JARs** to confirm.

Start YARN

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the YARN service.
 2. Select **Actions** > **Start**.

Deploy Client Configuration Files

1. On the Home page, click to the right of the cluster name and select **Deploy Client Configuration**.
2. Click the **Deploy Client Configuration** button in the confirmation pop-up that appears.

Upgrade the Spark Standalone Service

Required for all CDH upgrades:

1. Go to the Spark service.
2. If the Spark service is running, stop it:
 - a. Select **Actions** > **Stop** and click **Stop** to confirm.
3. Select **Actions** > **Install Spark JAR** and click **Install Spark JAR** to confirm.

Start Spark Services

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher

Note, all SPARK and SPARK2 are no longer differentiated, they are just SPARK.

1. Go to each Spark service.
2. Select **Actions** > **Start**.

Upgrade the Hive Metastore Database



Warning: Your upgrade will fail if you do not complete this step.

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the Hive service.
 2. If the Hive service is running, stop it:
 - a. Select **Actions** > **Stop** and click **Stop** to confirm.
 3. Select **Actions** > **Upgrade Hive Metastore Database Schema** and click **Upgrade Hive Metastore Database Schema** to confirm.
 4. If you have multiple instances of Hive, perform the upgrade on each metastore database.

5. Select **Actions** > **Validate Hive Metastore Schema** and click **Validate Hive Metastore Schema** to check that the schema is now valid.

Start Hive

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the Hive service.
 2. Select **Actions** > **Start**.

Validate the Hive Metastore Database Schema



Warning: Your upgrade will fail if you do not complete this step.

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Select **Actions** > **Validate Hive Metastore Schema** and click **Validate Hive Metastore Schema** to confirm.
 2. If you have multiple instances of Hive, perform the validation on each metastore database.
 3. Select **Actions** > **Validate Hive Metastore Schema** and click **Validate Hive Metastore Schema** to check that the schema is now valid.

Start Impala

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the Impala service.
 2. Select **Actions** > **Start**.

Upgrade Oozie

Required for the following upgrades:

- CDH 5.x to 6.0.0 or higher
1. Go to the Oozie service.
 2. Select **Actions** > **Stop** and click **Stop** to confirm.
 3. Select **Actions** > **Upgrade Oozie Database Schema** and click **Upgrade Oozie Database Schema** to confirm.

Upgrade the Oozie SharedLib

1. Go to the Oozie service
2. If the Oozie service is stopped, start it:
 - a. Select **Actions** > **Start** and click **Start** to confirm.
3. Select **Actions** > **Install Oozie SharedLib** and click **Install Oozie SharedLib** to confirm.

Start Remaining Cluster Services

1. Use rolling restart or full restart.
2. Ensure that all services are started or restarted. You can use Cloudera Manager to start the cluster, or you can restart the services individually. The Cloudera Manager Home page indicates which services have stale configurations and require restarting.
3. To start or restart the cluster:

- a. On the **Home > Status** page, click the down arrow to the right of the cluster name and select **Start** or **Restart**.
- 4. Click **Start** that appears in the next screen to confirm. The **Command Details** window shows the progress of starting services.
- 5. When **All services successfully started** appears, the task is complete and you can close the **Command Details** window.

Test the Cluster and Finalize HDFS Metadata

To determine if you can finalize the upgrade, run important workloads and ensure that they are successful. After you have finalized the upgrade, you cannot roll back to a previous version of HDFS without using backups. Verifying that you are ready to finalize the upgrade can take a long time.

When you are ready to finalize the upgrade, do one of the following:

- **If High Availability for HDFS is enabled:**
 1. Go to the HDFS service.
 2. Select **Actions > Finalize Rolling Upgrade** and click **Finalize Rolling Upgrade** to confirm.
- **If High Availability for HDFS is *not* enabled:**
 1. Go to the HDFS service.
 2. Click the **Instances** tab.
 3. Click the link for the **NameNode** instance. If you have enabled high availability for HDFS, click the link labeled **NameNode (Active)**.

The NameNode instance page displays.

 4. Select **Actions > Finalize Metadata Upgrade** and click **Finalize Metadata Upgrade** to confirm.

Troubleshooting CDH Upgrades

"Access denied" in install or update wizard

"Access denied" in install or update wizard during database configuration for Activity Monitor or Reports Manager.

Possible Reasons

Hostname mapping or permissions are not set up correctly.

Possible Solutions

- For hostname configuration, see [Configure Network Names](#).
- For permissions, make sure the values you enter into the wizard match those you used when you configured the databases. The value you enter into the wizard as the database hostname *must* match the value you entered for the hostname (if any) when you [configured the database](#).

For example, if you had entered the following when you created the database

```
grant all on activity_monitor.* TO 'amon_user'@'myhost1.myco.com' IDENTIFIED BY 'amon_password';
```

the value you enter here for the database hostname must be `myhost1.myco.com`. If you did not specify a host, or used a wildcard to allow access from any host, you can enter either the fully qualified domain name (FQDN), or `localhost`. For example, if you entered

```
grant all on activity_monitor.* TO 'amon_user'@'%' IDENTIFIED BY 'amon_password';
```

the value you enter for the database hostname can be either the FQDN or `localhost`.

Cluster hosts do not appear

Some cluster hosts do not appear when you click **Find Hosts** in install or update wizard.

Possible Reasons

You might have network connectivity problems.

Possible Solutions

- Make sure all cluster hosts have SSH port 22 open.
- Check other common causes of loss of connectivity such as firewalls and interference from SELinux.

Cannot start services after upgrade

You have upgraded the Cloudera Manager Server, but now cannot start services.

Possible Reasons

You might have mismatched versions of the Cloudera Manager Server and Agents.

Possible Solutions

Make sure you have upgraded the Cloudera Manager Agents on all hosts. (The previous version of the Agents will heartbeat with the new version of the Server, but you cannot start HDFS and MapReduce with this combination.)

HDFS DataNodes fail to start

After upgrading, HDFS DataNodes fail to start with exception:

```
Exception in secureMainjava.lang.RuntimeException: Cannot start datanode because the
configured max locked memory size (dfs.datanode.max.locked.memory) of 4294967296 bytes
is more than the datanode's available RLIMIT_MEMLOCK ulimit of 65536 bytes.
```

Possible Reasons

HDFS caching, which is enabled by default in CDH 5 and higher, requires new memlock functionality from Cloudera Manager Agents.

Possible Solutions

Do the following:

1. Stop all CDH and managed services.
2. On all hosts with Cloudera Manager Agents, hard-restart the Agents. Before performing this step, ensure you understand the semantics of the `hard_restart` command by reading [Hard Stopping and Restarting Agents](#).

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo sudo systemctl stop supervisord
sudo systemctl start cloudera-scm-agent
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent hard_restart
```

3. Start all services.

Cloudera services fail to start

Cloudera services fail to start.

Possible Reasons

Java might not be installed or might be installed at a custom location.

Possible Solutions

See [Configuring a Custom Java Home Location](#) for more information on resolving this issue.

Upgrading to CDH 5.8.0 or CDH 5.8.1 When Using the Flume Kafka Client

Due to the change of offset storage from ZooKeeper to Kafka in the CDH 5.8 Flume Kafka client, data might not be consumed by the Flume agents, or might be duplicated (if `kafka.auto.offset.reset=smallest`) during an upgrade to CDH 5.8.0 or CDH 5.8.1. To prevent this, perform the steps described below before you upgrade your system.



Important: This issue has been fixed for CDH 5.8.2 and higher. If you are upgrading to CDH 5.8.2 or higher, you do not need to perform this procedure. For more information, see [CDK Known Issues](#).

The upgrade process is based on this example configuration:

```
tier1.sources = source1
tier1.channels = channel1
tier1.sinks = sink1

tier1.sources.source1.type = org.apache.flume.source.kafka.KafkaSource
tier1.sources.source1.zookeeperConnect = zkhost:2181
tier1.sources.source1.topic = flumetopic
tier1.sources.source1.groupId = flume
tier1.sources.source1.channels = channel1
tier1.sources.source1.interceptors = i1 i2
tier1.sources.source1.interceptors.i1.type = timestamp
tier1.sources.source1.interceptors.i2.type = host
tier1.sources.source1.kafka.consumer.timeout.ms = 100

tier1.channels.channel1.type = org.apache.flume.channel.kafka.KafkaChannel
tier1.channels.channel1.brokerList=broker1:9092,broker2:9092
tier1.channels.channel1.zookeeperConnect=zkhost:2181
tier1.channels.channel1.topic=flumechannel1
tier1.channels.channel1.groupId = flumechannel
tier1.channels.channel1.capacity = 10000
tier1.channels.channel1.transactionCapacity = 1000

tier1.sinks.sink1.type = hdfs
tier1.sinks.sink1.hdfs.path = /tmp/kafka/%{topic}/
tier1.sinks.sink1.hdfs.filePrefix = %{host}-
tier1.sinks.sink1.hdfs.rollInterval = 60
tier1.sinks.sink1.hdfs.rollSize = 0
tier1.sinks.sink1.hdfs.rollCount = 0
tier1.sinks.sink1.hdfs.fileType = DataStream
tier1.sinks.sink1.channel = channel1
tier1.sinks.sink1.hdfs.kerberosKeytab = $KERBEROS_KEYTAB
tier1.sinks.sink1.hdfs.kerberosPrincipal = $KERBEROS_PRINCIPAL
```

Perform the following steps to upgrade CDH.

1. If you are using a version lower than CDH 5.7, first [upgrade to CDH 5.7](#). If for some reason you cannot upgrade to CDH 5.7, contact your Cloudera Sales Engineer for assistance, or file a support case with specific versions from which and to which you are upgrading.
2. Add the following sections to the source and channel for the Flume configuration:

```
# Added for source upgrade compatibility
tier1.sources.source1.kafka.bootstrap.servers = broker1:9092,broker2:9092
tier1.sources.source1.kafka.offsets.storage = kafka
tier1.sources.source1.kafka.dual.commit.enabled = true
tier1.sources.source1.kafka.consumer.group.id = flume
tier1.sources.source1.kafka.topics = flumetopic
```

```
# Added for channel upgrade compatability
tier1.channels.channell1.kafka.topic = flumechannell
tier1.channels.channell1.kafka.bootstrap.servers = broker1:9092,broker2:9092

tier1.channels.channell1.kafka.consumer.group.id = flumechannel
tier1.channels.channell1.kafka.offsets.storage = kafka
tier1.channels.channell1.kafka.dual.commit.enabled = true
```

3. Restart (or rolling restart) the Flume agents. This switches `offsets.storage` to Kafka, but keeps both the Kafka and ZooKeeper offsets updated because the `dual.commit.enabled` property is set to `true`. Confirm that Kafka messages are flowing through the Flume servers. Updating the offsets only occurs when new messages are consumed, so there must be at least one Kafka message consumed by the Flume agent, or one event passed through the Flume channel. Use the following commands to verify that Flume is properly updating the offsets in Kafka (the `egrep` command is used to match the correct topic names: in this example, `flumetopic` and `flumechannell`):

```
echo "exclude.internal.topics=false" > /tmp/consumer.config
kafka-console-consumer --consumer.config /tmp/consumer.config
--formatter "kafka.coordinator.GroupMetadataManager\$OffsetsMessageFormatter"

--zookeeper zkhost:2181 --topic __consumer_offsets |egrep -e
"flumetopic|flumechannell"
```

Output should be similar to the following and show that the Flume source and/or channel topics offsets are being incremented:

```
[flume,flumetopic,0]::[OffsetMetadata[70,cf9e5630-214e-4689-9869-5e077c936ffb],CommitTime
1469827951129,ExpirationTime 1469914351129]

[flumechannell,flumechannell,0]::[OffsetMetadata[61,875e7a82-1c22-43be-acaa-eb4d63e7f71e],CommitTime
1469827951128,ExpirationTime 1469914351128]

[flumechannell,flumechannell,0]::[OffsetMetadata[62,66bda888-0a70-4a02-a286-7e2e7d14050d],CommitTime
1469827951131,ExpirationTime 1469914351131]
```

4. Perform the upgrade to CDH 5.8.0 or CDH 5.8.1. The Flume agents are restarted during the process. Flume continues to consume the source topic where it left off, and the sinks continue draining from the Kafka channels where they left off. Post upgrade, remove the following deprecated properties from `flume.conf` because they are no longer used in CDH 5.8.0 or higher:

```
tier1.sources.source1.zookeeperConnect = zkhost:2181
tier1.sources.source1.topic = flumetopic
tier1.sources.source1.groupId = flume

tier1.channels.channell1.zookeeperConnect=zkhost:2181
tier1.channels.channell1.topic=flumechannell
tier1.channels.channell1.groupId = flumechannel
```

Rolling Back a CDH 5 to CDH 6 Upgrade

You can roll back an upgrade from CDH 5 to CDH 6. The rollback restores your CDH cluster to the state it was in before the upgrade, including Kerberos and TLS/SSL configurations.



Important: Any data created after the upgrade is lost.

In a typical upgrade, you first upgrade Cloudera Manager from version 5.x to version 6.x, and then you use the upgraded version of Cloudera Manager 6 to upgrade CDH 5 to CDH 6. (See [Upgrading the CDH Cluster](#) on page 92.) If you want to roll back this upgrade, follow these steps to roll back your cluster to its state prior to the upgrade.

You can roll back to CDH 5 after upgrading to CDH 6 only if the [HDFS upgrade has not been finalized](#). The rollback restores your CDH cluster to the state it was in before the upgrade, including Kerberos and TLS/SSL configurations.



Important: Follow all of the steps in the order presented in this topic. Cloudera recommends that you read through the backup and rollback steps before starting the backup process. You may want to create a detailed plan to help you anticipate potential problems.



Important:

These rollback steps depend on complete backups taken before upgrading Cloudera Manager and CDH. See [Backing Up Cloudera Manager](#) on page 41 and [Backing Up CDH](#) on page 73.

For steps where you need to restore the contents of a directory, clear the contents of the directory before copying the backed-up files to the directory. If you fail to do this, artifacts from the original upgrade can cause problems if you attempt the upgrade again after the rollback.

Review Limitations

The rollback procedure has the following limitations:

- **HDFS** – If you have finalized the HDFS upgrade, *you cannot roll back your cluster*.
- **Configuration changes**, including the addition of new services or roles after the upgrade, are not retained after rolling back Cloudera Manager.

Cloudera recommends that you not make configuration changes or add new services and roles until you have finalized the HDFS upgrade and no longer require the option to roll back your upgrade.

- **HBase** – If your cluster is configured to use HBase replication, data written to HBase after the upgrade might not be replicated to peers when you start your rollback. This topic does not describe how to determine which, if any, peers have the replicated data and how to roll back that data. For more information about HBase replication, see [HBase Replication](#).
- **Sqoop 1** – Because of the changes introduced in Sqoop metastore logic, the metastore database that is created by the CDH 6.x version of Sqoop cannot be used by earlier versions.
- **Sqoop 2** – As described in the upgrade process, Sqoop2 had to be stopped and deleted before the upgrade process and therefore will not be available after the rollback.
- **Kafka** – Once the Kafka log format and protocol version configurations (the `inter.broker.protocol.version` and `log.message.format.version` properties) are set to the new version (or left blank, which means to use the latest version), *Kafka rollback is not possible*.

Stop the Cluster

1. On the **Home > Status** tab, click



to the right of the cluster name and select **Stop**.

2. Click **Stop** in the confirmation screen. The **Command Details** window shows the progress of stopping services.

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

(Parcels) Downgrade the Software

Follow these steps only if your cluster was upgraded using Cloudera *parcels*.

1. Log in to the Cloudera Manager Admin Console.
2. Select **Hosts > Parcels**.

A list of parcels displays.

3. Locate the CDH 5 parcel and click **Activate**. (This automatically deactivates the CDH 6 parcel.) See [Activating a Parcel](#) for more information. If the parcel is not available, use the **Download** button to download the parcel.
4. If you include any additional components in your cluster, such as Search or Impala, click **Activate** for those parcels.



Important:

Do not start any services. (Select the **Activate Only** option.)

If you accidentally restart services, stop your cluster before proceeding.

Stop Cloudera Manager

1. Stop the **Cloudera Management Service**.
 - a. Log in to the Cloudera Manager Admin Console.
 - b. Select **Clusters > Cloudera Management Service**.
 - c. Select **Actions > Stop**.

2. Stop the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server stop
```

3. Hard stop the Cloudera Manager agents. Run the following command on all hosts:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop supervisord
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent hard_stop
```

(Packages) Downgrade the Software

Follow these steps only if your cluster was upgraded using *packages*.

Run Package Commands

1. Log in as a privileged user to all hosts in your cluster.
2. Back up the repository directory. You can create a top-level backup directory and an environment variable to reference the directory using the following commands. You can also substitute another directory path in the backup commands below:

```
export CM_BACKUP_DIR="`date +%F`-CM"
mkdir -p $CM_BACKUP_DIR
```

RHEL / CentOS

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/yum.repos.d
```

SLES

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/zypp/repos.d
```

Debian / Ubuntu

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/apt/sources.list.d
```

3. Restore the CDH 5 repository directory from the backup taken before upgrading to CDH 6. For example:

```
tar -xf CM6CDH5/repository.tar -C CM6CDH5/
```

RHEL

```
rm -rf /etc/yum.repos.d/*
cp -rp CM6CDH5/etc/yum.repos.d/* /etc/yum.repos.d/
```

SLES

```
rm -rf /etc/zypp/repos.d
cp -rp CM6CDH5/etc/zypp/repos.d/* /etc/zypp/repos.d/
```

Debian / Ubuntu

```
rm -rf /etc/apt/sources.list.d/*
cp -rp CM6CDH5/etc/apt/sources.list.d/* /etc/apt/sources.list.d/
```

4. Run the following command to uninstall CDH 6 and reinstall the CDH 5 packages:**RHEL / CentOS**

```
sudo yum clean all
```

```
sudo yum remove avro-tools flume-ng avro-libs hadoop-hdfs-fuse hadoop-hdfs-nfs3
hadoop-httpfs hadoop-kms hbase-solr hive-hbase hive-webhcat hue impala impala-shell
kafka kite kudu oozie pig search sentry sentry-hdfs-plugin solr-crunch solr-mapreduce
spark-core spark-python sqoop zookeeper parquet hbase solr
```

```
sudo yum -y install --setopt=timeout=180 bigtop-utils solr-doc oozie-client hue-spark
kite crunch-doc sqoop hue-rdbms hbase-solr hue-plugins pig spark-python oozie hadoop-kms
bigtop-tomcat hbase hue-sqoop sqoop2 spark-core hadoop-mapreduce avro-tools hadoop-hdfs
avro-libs hadoop sqoop2-client mahout avro-doc hue-impala hbase-solr-doc hive-jdbc
crunch zookeeper hadoop-hdfs-nfs3 bigtop-jsvc hue-common hue-hbase hadoop-client
hive-webhcat parquet-format hue-beeswax keytrustee-keyprovider hue-pig llama hive-hcatalog
kudu kafka solr hue-search hive-hbase search solr-crunch flume-ng hadoop-httpfs
hue-security sentry hive sentry-hdfs-plugin hadoop-yarn hadoop-hdfs-fuse parquet
hadoop-0.20-mapreduce impala-shell impala hue-zookeeper solr-mapreduce
```

SLES

```
sudo zypper clean --all
```

```
sudo zypper remove avro-tools flume-ng avro-libs hadoop-hdfs-fuse hadoop-hdfs-nfs3
hadoop-httpfs hadoop-kms hbase-solr hive-hbase hive-webhcat hue impala impala-shell
```

```
kafka kite kudu oozie pig search sentry sentry-hdfs-plugin solr-crunch solr-mapreduce  
spark-core spark-python sqoop zookeeper parquet hbase solr
```

```
sudo zypper install solr-doc oozie-client hue-spark kite crunch-doc sqoop hue-rdbms  
hbase-solr hue-plugins pig spark-python oozie hadoop-kms bigtop-tomcat hbase hue-sqoop  
sqoop2 spark-core hadoop-mapreduce avro-tools hadoop-hdfs avro-libs hadoop sqoop2-client  
mahout avro-doc hue-impala hbase-solr-doc hive-jdbc crunch zookeeper hadoop-hdfs-nfs3  
bigtop-jsvc hue-common hue-hbase hadoop-client hive-webhcat parquet-format hue-beeswax  
keytrustee-keyprovider hue-pig llama hive-hcatalog kudu kafka solr hue-search hive-hbase  
search solr-crunch flume-ng hadoop-httpfs hue-security sentry hive sentry-hdfs-plugin  
hadoop-yarn hadoop-hdfs-fuse parquet hadoop-0.20-mapreduce impala-shell impala  
hue-zookeeper solr-mapreduce
```

Debian / Ubuntu

```
sudo apt-get update  
sudo apt-get remove avro-tools flume-ng avro-libs hadoop-hdfs-fuse hadoop-hdfs-nfs3  
hadoop-httpfs hadoop-kms hbase-solr hive-hbase hive-webhcat hue impala impala-shell  
kafka kite kudu oozie pig search sentry sentry-hdfs-plugin solr-crunch solr-mapreduce  
spark-core spark-python sqoop zookeeper parquet hbase solr
```

```
sudo apt-get update  
sudo apt-get install solr-doc oozie-client hue-spark kite crunch-doc sqoop hue-rdbms  
hbase-solr hue-plugins pig spark-python oozie hadoop-kms bigtop-tomcat hbase hue-sqoop  
sqoop2 spark-core hadoop-mapreduce avro-tools hadoop-hdfs avro-libs hadoop sqoop2-client  
mahout avro-doc hue-impala hbase-solr-doc hive-jdbc crunch zookeeper hadoop-hdfs-nfs3  
bigtop-jsvc hue-common hue-hbase hadoop-client hive-webhcat parquet-format hue-beeswax  
keytrustee-keyprovider hue-pig llama hive-hcatalog kudu kafka solr hue-search hive-hbase  
search solr-crunch flume-ng hadoop-httpfs hue-security sentry hive sentry-hdfs-plugin  
hadoop-yarn hadoop-hdfs-fuse parquet hadoop-0.20-mapreduce impala-shell impala  
hue-zookeeper solr-mapreduce
```

Restore Cloudera Manager Databases

Restore the Cloudera Manager databases from the backup of Cloudera Manager that was taken before upgrading the cluster to CDH 6. See the procedures provided by your database vendor.

- **MariaDB 5.5:** <http://mariadb.com/kb/en/mariadb/backup-and-restore-overview/>
- **MySQL 5.5:** <http://dev.mysql.com/doc/refman/5.5/en/backup-and-recovery.html>
- **MySQL 5.6:** <http://dev.mysql.com/doc/refman/5.6/en/backup-and-recovery.html>
- **PostgreSQL 8.4:** <https://www.postgresql.org/docs/8.4/static/backup.html>
- **PostgreSQL 9.2:** <https://www.postgresql.org/docs/9.2/static/backup.html>
- **PostgreSQL 9.3:** <https://www.postgresql.org/docs/9.3/static/backup.html>
- **Oracle 11gR2:** http://docs.oracle.com/cd/E11882_01/backup.112/e10642/toc.htm
- **HyperSQL:** http://hsqldb.org/doc/guide/management-chapt.html#mtc_backup

Restore Cloudera Manager Server

Use the backup of CDH that was taken before the upgrade to restore Cloudera Manager Server files and directories. Substitute the path to your backup directory for `cm6_cdh5` in the following steps:

1. On the host where the Event Server role is configured to run, restore the Events Server directory from the CM 6/CDH 5 backup.

```
rm -rf /var/lib/cloudera-scm-eventserver/*  
cp -rp /var/lib/cloudera-scm-eventserver_cm6_cdh5/* /var/lib/cloudera-scm-eventserver/
```

2. Remove the Agent runtime state. Run the following command on all hosts:

```
rm -rf /var/run/cloudera-scm-agent /var/lib/cloudera-scm-agent/response.avro
```


This command may return a message similar to: `rm: cannot remove
'/var/run/cloudera-scm-agent/process': Device or resource busy`. You can ignore this message.

3. On the host where the Service Monitor is running, restore the Service Monitor directory:

```
rm -rf /var/lib/cloudera-service-monitor/*
cp -rp /var/lib/cloudera-service-monitor_cm6_cdh5/* /var/lib/cloudera-service-monitor/
```

4. On the host where the Host Monitor is running, restore the Host Monitor directory:

```
rm -rf /var/lib/cloudera-host-monitor/*
cp -rp /var/lib/cloudera-host-monitor_cm6_cdh5/* /var/lib/cloudera-host-monitor/
```

5. Restore the Cloudera Navigator storage directory from the CM 6/CDH 5 backup.

```
rm -rf /var/lib/cloudera-scm-navigator/*
cp -rp /var/lib/cloudera-scm-navigator_cm6_cdh5/* /var/lib/cloudera-scm-navigator/
```

Start Cloudera Manager

1. Log in to the Cloudera Manager Server host.

```
ssh my_cloudera_manager_server_host
```

2. Start the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server
```

If the Cloudera Manager Server starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server start
```

You should see the following:

```
Starting cloudera-scm-server: [ OK ]
```

3. Start the **Cloudera Manager Agent**.

Run the following commands on all cluster hosts:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-agent
```

If the agent starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent start
```

You should see the following:

```
Starting cloudera-scm-agent: [ OK ]
```

4. Start the **Cloudera Management Service**.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters > Cloudera Management Service**.

c. Select **Actions** > **Start**.

Roll Back ZooKeeper

1. Using the backup of ZooKeeper that you created when backing up your CDH 5.x cluster, restore the contents of the `dataDir` on each ZooKeeper server. These files are located in a directory specified with the `dataDir` property in the ZooKeeper configuration. The default location is `/var/lib/zookeeper`. For example:

```
rm -rf /var/lib/zookeeper/*
cp -rp /var/lib/zookeeper_cm6_cdh5/* /var/lib/zookeeper/
```

2. Make sure that the permissions of all the directories and files are as they were before the upgrade.
3. Start ZooKeeper using Cloudera Manager.

Roll Back HDFS

You cannot roll back HDFS while high availability is enabled. The rollback procedure in this topic creates a temporary configuration without high availability. Regardless of whether high availability is enabled, follow the steps in this section.

1. Roll back all of the **Journal Nodes**. (Only required for clusters where high availability is enabled for HDFS). Use the [JournalNode backup you created](#) when you backed up HDFS before upgrading to CDH 6.

- a. Log in to each **Journal Node** host and run the following commands:

```
rm -rf /dfs/jn/ns1/current/*
```

```
cp -rp /dfs/jn/ns1/previous/* /dfs/jn/ns1/current/
```

- b. Start the JournalNodes using Cloudera Manager:

- a. Go to the HDFS service.
- b. Select the **Instances** tab.
- c. Select all JournalNode roles from the list.
- d. Click **Actions for Selected** > **Start**.

2. Roll back all of the **NameNodes**. Use the [NameNode backup directory](#) you created before upgrading CDH (`/etc/hadoop/conf.rollback.namenode`) to perform the following steps on all NameNode hosts:

- a. (Clusters with TLS enabled only) Edit the `/etc/hadoop/conf.rollback.namenode/ssl-server.xml` file on all NameNode hosts (located in the temporary rollback directory) and update the keystore passwords with the actual cleartext passwords.

The passwords will have values that look like this:

```
<property>
  <name>ssl.server.keystore.password</name>
  <value>*****</value>
</property>
<property>
  <name>ssl.server.keystore.keypassword</name>
  <value>*****</value>
</property>
```

- b. (TLS only) Edit the `/etc/hadoop/conf.rollback.namenode/ssl-server.xml` file and remove the `hadoop.security.credential.provider.path` property.

3. Edit the `/etc/hadoop/conf.rollback.namenode/hdfs-site.xml` file on all NameNode hosts and make the following changes:

- a. Delete the `cloudera.navigator.client.config` property.
- b. Delete the `dfs.namenode.audit.loggers` property.

- c. Change the path in the `dfs.hosts` property to the value shown in the example below. The file name, `dfs_all_hosts.txt`, may have been changed by a user. If so, substitute the correct file name.

```
# Original version of the dfs.hosts property:
<property>
<name>dfs.hosts</name>
<value>/var/run/cloudera-scm-agent/process/63-hdfs-NAMENODE/dfs_all_hosts.txt</value>
</property>
```

```
# New version of the dfs.hosts property:
<property>
<name>dfs.hosts</name>
<value>/etc/hadoop/conf.rollback.namenode/dfs_all_hosts.txt</value>
</property>
```

- d. Edit the `/etc/hadoop/conf.rollback.namenode/core-site.xml` and change the value of the `net.topology.script.file.name` property to `/etc/hadoop/conf.rollback.namenode`. For example:

```
# Original property
<property>
<name>net.topology.script.file.name</name>
<value>/var/run/cloudera-scm-agent/process/63-hdfs-NAMENODE/topology.py</value>
</property>
```

```
# New property
<property>
<name>net.topology.script.file.name</name>
<value>/etc/hadoop/conf.rollback.namenode/topology.py</value>
</property>
```

- e. Edit the `/etc/hadoop/conf.rollback.namenode/topology.py` file and change the value of `MAP_FILE` to `/etc/hadoop/conf.rollback.namenode`. For example:

```
MAP_FILE = '/etc/hadoop/conf.rollback.namenode/topology.map'
```

- f. (TLS-enabled clusters only) Run the following command:

```
sudo -u hdfs kinit hdfs/<NameNode Host name> -l 7d -kt
/etc/hadoop/conf.rollback.namenode/hdfs.keytab
```

- g. Run the following command:

```
sudo -u hdfs hdfs --config /etc/hadoop/conf.rollback.namenode namenode -rollback
```

- h. Restart the NameNodes and JournalNodes using Cloudera Manager:

- Go to the HDFS service.
- Select the **Instances** tab, and then select all NameNode and JournalNode roles from the list.
- Click **Actions for Selected > Restart**.

4. Rollback the DataNodes.

Use the [DataNode rollback directory](#) you created before upgrading CDH (`/etc/hadoop/conf.rollback.datanode`) to perform the following steps on *all DataNode hosts*:

- (Clusters with TLS enabled only) Edit the `/etc/hadoop/conf.rollback.datanode/ssl-server.xml` file on all DataNode hosts (Located in the temporary rollback directory.) and update the keystore passwords (`ssl.server.keystore.password` and `ssl.server.keystore.keypassword`) with the actual passwords.

The passwords will have values that look like this:

```
<property>
  <name>ssl.server.keystore.password</name>
  <value>*****</value>
</property>
<property>
  <name>ssl.server.keystore.keypassword</name>
  <value>*****</value>
</property>
```

2. (TLS only) Edit the `/etc/hadoop/conf.rollback.datanode/ssl-server.xml` file and remove the `hadoop.security.credential.provider.path` property.
3. Edit the `/etc/hadoop/conf.rollback.datanode/hdfs-site.xml` file and remove the `dfs.datanode.max.locked.memory` property.
4. Run the following command:

```
cd /etc/hadoop/conf.rollback.datanode
sudo -u hdfs hdfs --config /etc/hadoop/conf.rollback.datanode datanode -rollback
```

After rolling back the DataNodes, terminate the console session by typing **Control-C**.

5. If High Availability for HDFS is enabled, restart the HDFS service. In the Cloudera Manager Admin Console, go to the HDFS service and select **Actions > Restart**.
6. If high availability is not enabled for HDFS, use the Cloudera Manager Admin Console to restart all NameNodes and DataNodes.
 - a. Go to the HDFS service.
 - b. Select the **Instances** tab
 - c. Select all DataNode and NameNode roles from the list.
 - d. Click **Actions for Selected > Restart**.

5. If high availability is not enabled for HDFS, roll back the **Secondary NameNode**.

- a. (Clusters with TLS enabled only) Edit the `/etc/hadoop/conf.rollback.secondarynamenode/ssl-server.xml` file on all Secondary NameNode hosts (Located in the temporary rollback directory.) and update the keystore passwords with the actual cleartext passwords.

The passwords will have values that look like this:

```
<property>
  <name>ssl.server.keystore.password</name>
  <value>*****</value>
</property>
<property>
  <name>ssl.server.keystore.keypassword</name>
  <value>*****</value>
</property>
```

- b. (TLS only) Edit the `/etc/hadoop/conf.rollback.secondarynamenode/ssl-server.xml` file and remove the `hadoop.security.credential.provider.path` property.
- c. Log in to the Secondary NameNode host and run the following commands:

```
rm -rf /dfs/snn/*
cd /etc/hadoop/conf.rollback.secondarynamenode/
sudo -u hdfs hdfs --config /etc/hadoop/conf.rollback.secondarynamenode secondarynamenode
-format
```

6. Restart the HDFS service. Open the Cloudera Manager Admin Console, go to the HDFS service page, and select **Actions > Restart**.

The **Restart Command** page displays the progress of the restart. Wait for the page to display the **Successfully restarted service** message before continuing.

Start the Key Management Server

Restart the Key Management Server. Open the Cloudera Manager Admin Console, go to the KMS service page, and select **Actions > Start**.

Start the HBase Service

Restart the HBase Service. Open the Cloudera Manager Admin Console, go to the HBase service page, and select **Actions > Start**.

If you encounter errors when starting HBase, delete the znode in ZooKeeper and then start HBase again:

1. In Cloudera Manager, look up the value of the `zookeeper.znode.parent` property. The default value is `/hbase`.
2. Connect to the ZooKeeper ensemble by running the following command from any HBase gateway host:

```
zookeeper-client -server zookeeper_ensemble
```

To find the value to use for `zookeeper_ensemble`, open the `/etc/hbase/conf.cloudera.<HBase service name>/hbase-site.xml` file on any HBase gateway host. Use the value of the `hbase.zookeeper.quorum` property.



Note:

If you have deployed a secure cluster, you must connect to ZooKeeper using a client `jaas.conf` file. You can find such a file in an HBase process directory (`/var/run/cloudera-scm-agent/process/`). Specify the `jaas.conf` using the JVM flags by running the following commands in the ZooKeeper client:

```
CLIENT_JVMFLAGS=
"-Djava.security.auth.login.config=/var/run/cloudera-scm-agent/process/HBase_process_directory/jas.conf"
zookeeper-client -server <zookeeper_ensemble>
```

The ZooKeeper command-line interface opens.

3. Enter the following command:

```
rmr /hbase
```

Restore CDH Databases

Restore the following databases from the CDH 5 backups:

- Hive Metastore
- Hue
- Oozie
- Sentry Server

The steps for backing up and restoring databases differ depending on the database vendor and version you select for your cluster and are beyond the scope of this document.



Important: Restore the databases to their exact state as of when you took the backup. Do not merge in any changes that may have occurred during the subsequent upgrade.

See the following vendor resources for more information:

- **MariaDB 5.5:** <http://mariadb.com/kb/en/mariadb/backup-and-restore-overview/>
- **MySQL 5.5:** <http://dev.mysql.com/doc/refman/5.5/en/backup-and-recovery.html>
- **MySQL 5.6:** <http://dev.mysql.com/doc/refman/5.6/en/backup-and-recovery.html>
- **PostgreSQL 8.4:** <https://www.postgresql.org/docs/8.4/static/backup.html>
- **PostgreSQL 9.2:** <https://www.postgresql.org/docs/9.2/static/backup.html>
- **PostgreSQL 9.3:** <https://www.postgresql.org/docs/9.3/static/backup.html>
- **Oracle 11gR2:** http://docs.oracle.com/cd/E11882_01/backup.112/e10642/toc.htm

Start the Sentry Service

1. Log in to the Cloudera Manager Admin Console.
2. Go to the **Sentry** service.
3. Click **Actions** > **Start**.

Roll Back Cloudera Search

1. Start the HDFS, Zookeeper and Sentry services.
2. Re-initialize the configuration metadata in Zookeeper by running the following commands:

```
export ZKCLI_JVM_FLAGS="-Djava.security.auth.login.config=~/.solr-jaas.conf  
-DzkACLProvider=org.apache.solr.common.cloud.ConfigAwareSaslZkACLProvider"
```

```
sudo -u solr mkdir /tmp/c5-config-backup
```

```
sudo -u solr chmod 755 /tmp/c5-config-backup
```

```
sudo -u solr hdfs dfs -copyToLocal /user/solr/upgrade_backup/zk_backup/*  
/tmp/c5-config-backup
```

```
/opt/cloudera/cm/solr-upgrade/solr-rollback.sh zk-meta -c /tmp/c5-config-backup
```

3. Re-initialize configuration metadata in the local file system:

- On each host configured with SOLR_SERVER role, run the following commands:

```
rm -rf <solr_data_directory>/*
```

2. The value of `<solr_data_directory>` is configured via CM parameter named "Solr Data Directory" (the default is `/var/lib/solr`).
- Inspect the sub-directories present inside `<backup_location>/localfs_backup` directory (where `<backup_location>` is the value configured as part of "Upgrade Backup Directory" configuration parameter for Solr in CM). For each of the sub-directories:
 1. The sub-directory name refers to the internal `role_id` of the Solr server on a particular host in Cloudera Manager. Identify the corresponding hostname by querying Cloudera Manager database. To find the `role_id`:
 - a. Log in to the Cloudera Manager Admin Console.
 - b. Go to the HDFS File browser.
 - c. Open the `solr/upgrade_backup/localfs_backup` file. The `role_id` is within this file.
 2. Copy the contents of this sub-directory on the identified host (e.g. H1) at location specified by "Solr Data Directory" parameter in CM. The default value for this parameter is `/var/lib/solr`

- Login to host H1.
- Run the following command:

```
sudo -u solr hdfs dfs -copyToLocal /user/solr/upgrade_backup/localfs_backup/<role_id> /var/lib/solr
```

4. Start the Solr service.

Roll Back Hue

1. Restore the file, `app.reg`, from your backup:

- **Parcel installations**

```
rm -rf /opt/cloudera/parcels/CDH/lib/hue/app.reg
cp -rp app.reg_cm5_cdh5_backup /opt/cloudera/parcels/CDH/lib/hue/app.reg
```

- **Package Installations**

```
rm -rf /usr/lib/hue/app.reg
cp -rp app.reg_cm5_cdh5_backup /usr/lib/hue/app.reg
```

Roll Back Kafka

A CDH 6 cluster that is running Kafka can be rolled back to the previous CDH5/CDK versions as long as the `inter.broker.protocol.version` and `log.message.format.version` properties have not been set to the new version or removed from the configuration.

To perform the rollback using Cloudera Manager:

1. Activate the previous CDK parcel. Please note, that when rolling back Kafka from CDH 6 to CDH 5/CDK, the Kafka cluster will restart. Rolling restart is not supported for this scenario. See [Activating a Parcel](#).
2. Remove the following properties from the **Kafka Broker Advanced Configuration Snippet (Safety Valve)** configuration property.
 - `inter.broker.protocol.version`
 - `log.message.format.version`

Roll Back Sqoop 2

Upgrading to CDH 6.x required you to delete the Sqoop 2 service before upgrading. To roll back your Sqoop 2 service:

1. Add the Sqoop 2 service using Cloudera Manager.
2. Restore the Sqoop 2 database from your backup. [See the documentation for your database](#) for details.

If you are not using the default embedded Derby database for Sqoop 2, [restore the database](#) you have configured for Sqoop 2. Otherwise, restore the `repository` subdirectory of the Sqoop 2 metastore directory from your backup. This location is specified with the **Sqoop 2 Server Metastore Directory** property. The default location is `/var/lib/sqoop2`. For this default location, Derby database files are located in `/var/lib/sqoop2/repository`.

Deploy the Client Configuration

1. On the **Home > Status** tab, click



to the right of the cluster name and select **Deploy Client Configuration**.

2. Click **Deploy Client Configuration**.

Restart the Cluster

1. On the **Home > Status** tab, click



to the right of the cluster name and select **Restart**.

2. Click **Restart** that appears in the next screen to confirm. If you have enabled [high availability for HDFS](#), you can choose [Rolling Restart](#) instead to minimize cluster downtime. The **Command Details** window shows the progress of stopping services.

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

Roll Back Cloudera Navigator Encryption Components

If you are rolling back any encryption components (Key Trustee Server, Key Trustee KMS, HSM KMS, Key HSM, or Navigator Encrypt), first refer to:

- [Backing Up and Restoring Key Trustee Server and Clients](#)
- [HSM KMS High Availability Backup and Recovery](#)
- [Manually Backing Up Navigator Encrypt](#)

Roll Back Key Trustee Server



Note: If rolling back multiple encryption product components, it is recommended that you begin with the Key Trustee Server.

To roll back Key Trustee Server, replace the currently used parcel (for example, the parcel for version 6.0.0) with the parcel for the version to which you wish to roll back (for example, version 5.14.0). See [Parcels](#) for detailed instructions on using parcels.

Roll Back Key HSM

To roll back Key HSM:

1. **Install the version of Navigator Key HSM to which you wish to roll back**

Install the Navigator Key HSM package using `yum`:

```
sudo yum downgrade keytrustee-keyhsm
```

Cloudera Navigator Key HSM is installed to the `/usr/share/keytrustee-server-keyhsm` directory by default.

2. **Rename Previously-Created Configuration Files**

For Key HSM major version rollbacks, previously-created configuration files do not authenticate with the HSM and Key Trustee Server, so you must recreate these files by re-executing the `setup` and `trust` commands. First, navigate to the Key HSM installation directory and rename the `applications.properties`, `keystore`, and `truststore` files:

```
cd /usr/share/keytrustee-server-keyhsm/
mv application.properties application.properties.bak
mv keystore keystore.bak
mv truststore truststore.bak
```

3. **Initialize Key HSM**

Run the `service keyhsm setup` command in conjunction with the name of the target HSM distribution:

```
sudo service keyhsm setup [keysecure|thales|luna]
```

For more details, see [Initializing Navigator Key HSM](#).

4. Establish Trust Between Key HSM and the Key Trustee Server

The Key HSM service must explicitly trust the Key Trustee Server certificate (presented during TLS handshake). To establish this trust, run the following command:

```
sudo keyhsm trust /path/to/key_trustee_server/cert
```

For more details, see [Establish Trust from Key HSM to Key Trustee Server](#).

5. Start the Key HSM Service

Start the Key HSM service:

```
sudo service keyhsm start
```

6. Establish Trust Between Key Trustee Server and Key HSM

Establish trust between the Key Trustee Server and the Key HSM by specifying the path to the private key and certificate:

```
sudo ktadmin keyhsm --server https://keyhsm01.example.com:9090 \
--client-certfile /etc/pki/cloudera/certs/mycert.crt \
--client-keyfile /etc/pki/cloudera/certs/mykey.key --trust
```

For a password-protected Key Trustee Server private key, add the `--passphrase` argument to the command (enter the password when prompted):

```
sudo ktadmin keyhsm --passphrase \
--server https://keyhsm01.example.com:9090 \
--client-certfile /etc/pki/cloudera/certs/mycert.crt \
--client-keyfile /etc/pki/cloudera/certs/mykey.key --trust
```

For additional details, see [Integrate Key HSM and Key Trustee Server](#).

7. Remove Configuration Files From Previous Installation

After completing the rollback, remove the saved configuration files from the previous installation:

```
cd /usr/share/keytrustee-server-keyhsm/
rm application.properties.bak
rm keystore.bak
rm truststore.bak
```

Roll Back Key Trustee KMS Parcels

To roll back Key Trustee KMS parcels, replace the currently used parcel (for example, the parcel for version 6.0.0) with the parcel for the version to which you wish to roll back (for example, version 5.14.0). See [Parcels](#) for detailed instructions on using parcels.

Roll Back Key Trustee KMS Packages

To roll back Key Trustee KMS packages:

1. After [Setting Up an Internal Repository](#) on page 170 configure the Key Trustee KMS host to use the repository. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.



Note: For downgrade, you must specify that your internal repository use CDH 5 packages rather than CDH 6 packages. See [Configuring a Local Package Repository](#) on page 184.

2. Downgrade the `keytrustee-provider` package using the appropriate command for your operating system:

RHEL-compatible

```
sudo yum downgrade keytrustee-keyprovider
```

Roll Back HSM KMS Parcels

To roll back the HSM KMS parcels, replace the currently used parcel (for example, the parcel for version 6.0.0) with the parcel for the version to which you wish to roll back (for example, version 5.14.0). See [Parcels](#) for detailed instructions on using parcels.

See [Upgrading HSM KMS Using Packages](#) on page 172 for detailed instructions on using packages.

Roll Back HSM KMS Packages

To roll back HSM KMS packages:

1. After [Setting Up an Internal Repository](#) on page 170 configure the HSM KMS host to use the repository. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.



Note: For downgrade, you must specify that your internal repository use CDH 5 packages rather than CDH 6 packages. See [Configuring a Local Package Repository](#) on page 184.

2. Downgrade the `keytrustee-provider` package using the appropriate command for your operating system:

RHEL-compatible

```
sudo yum downgrade keytrustee-keyprovider
```

Roll Back Navigator Encrypt

To roll back Cloudera Navigator Encrypt:

1. If you have configured and are using an RSA master key file with OAEP padding, then you must revert this setting to its original value:

```
# navencrypt key --change
```

2. Stop the Navigator Encrypt mount service:

```
$ sudo /etc/init.d/navencrypt-mount stop
```

3. To fully downgrade Navigator Encrypt, manually downgrade all of the associated Navigator Encrypt packages (in the order listed):

- a. `navencrypt`
- b. `navencrypt-kernel-module`
- c. `cloudera-navencryptfs-kmp-<kernel_flavor>` for SLES



Note: Replace `kernel_flavor` with the kernel flavor for your system. Navigator Encrypt supports the `default`, `xen`, and `ec2` kernel flavors.

- d. `libkeytrustee`

4. Restart the Navigator Encrypt mount service:

```
$ sudo /etc/init.d/navencrypt-mount start
```

(Optional) Cloudera Manager Rollback Steps

After you complete the rollback steps, your cluster is using Cloudera Manager 6 to manage your CDH 5 cluster. You can continue to use Cloudera Manager 6 to manage your CDH 5 cluster, or you can downgrade to Cloudera Manager 5 by following these steps:

Stop Cloudera Manager

1. Stop the **Cloudera Management Service**.

- a. Log in to the Cloudera Manager Admin Console.
- b. Select **Clusters** > **Cloudera Management Service**.
- c. Select **Actions** > **Stop**.

2. Stop the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop cloudera-scm-server
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server stop
```

3. Hard stop the Cloudera Manager agents. Run the following command on all hosts:

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl stop supervisord
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent hard_stop
```

4. Back up the repository directory. You can create a top-level backup directory and an environment variable to reference the directory using the following commands. You can also substitute another directory path in the backup commands below:

```
export CM_BACKUP_DIR="`date +%F`-CM"
mkdir -p $CM_BACKUP_DIR
```

5. Back up the existing repository directory.

RHEL / CentOS

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/yum.repos.d
```

SLES

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/zypp/repos.d
```

Debian / Ubuntu

```
sudo -E tar -cf $CM_BACKUP_DIR/repository.tar /etc/apt/sources.list.d
```

Restore the Cloudera Manager 5 Repository Files

Copy the repository directory from the backup taken before upgrading to Cloudera Manager 6.x.

```
rm -rf /etc/yum.repos.d/*
cp -rp /etc/yum.repos.d_cm5cdh5/* /etc/yum.repos.d/
```

Restore Packages

1. Run the following commands on all hosts:

Operating System	Command
RHEL	<pre>sudo yum remove cloudera-manager-daemons cloudera-manager-agent</pre>
	<pre>sudo yum clean all sudo yum install cloudera-manager-agent</pre>
SLES	<pre>sudo zypper remove cloudera-manager-daemons cloudera-manager-agent</pre>
	<pre>sudo zypper refresh -s sudo zypper install cloudera-manager-agent</pre>
Ubuntu or Debian	<pre>sudo apt-get purge cloudera-manager-daemons cloudera-manager-agent</pre>
	<pre>sudo apt-get update sudo apt-get install cloudera-manager-agent</pre>

2. Run the following commands on the Cloudera Manager server host:

Operating System	Command
RHEL	<code>sudo yum remove cloudera-manager-server</code>
	<code>sudo yum install cloudera-manager-server</code>
SLES	<code>sudo zypper remove cloudera-manager-server</code>
	<code>sudo zypper install cloudera-manager-server</code>
Ubuntu or Debian	<code>sudo apt-get purge cloudera-manager-server</code>
	<code>sudo apt-get install cloudera-manager-server</code>

Restore Cloudera Manager Databases

Restore the Cloudera Manager databases from the backup of Cloudera Manager that was taken before upgrading to Cloudera Manager 6. See the procedures provided by your database vendor.

These databases include the following:

- Cloudera Manager Server
- Reports Manager
- Navigator Audit Server
- Navigator Metadata Server
- Activity Monitor (Only used for MapReduce 1 monitoring).
- **MariaDB 5.5:** <http://mariadb.com/kb/en/mariadb/backup-and-restore-overview/>
- **MySQL 5.5:** <http://dev.mysql.com/doc/refman/5.5/en/backup-and-recovery.html>
- **MySQL 5.6:** <http://dev.mysql.com/doc/refman/5.6/en/backup-and-recovery.html>
- **PostgreSQL 8.4:** <https://www.postgresql.org/docs/8.4/static/backup.html>
- **PostgreSQL 9.2:** <https://www.postgresql.org/docs/9.2/static/backup.html>
- **PostgreSQL 9.3:** <https://www.postgresql.org/docs/9.3/static/backup.html>
- **Oracle 11gR2:** http://docs.oracle.com/cd/E11882_01/backup.112/e10642/toc.htm
- **HyperSQL:** http://hsqldb.org/doc/guide/management-chapt.html#mtc_backup

Here is an sample command to restore a MySQL database:

```
mysql -u username -ppassword --host=hostname cm < backup.sql
```

Restore Cloudera Manager Server

Use the backup of Cloudera Manager 5.x taken before upgrading to Cloudera Manager 6.x for the following steps:

1. If you used the backup commands provided in [Backing Up Cloudera Manager](#) on page 41, extract the Cloudera Manager 5 backup archives you created:

```
tar -xf CM5CDH5/cloudera-scm-agent.tar -C CM5CDH5/
tar -xf CM5CDH5/cloudera-scm-server.tar -C CM5CDH5/
```

2. On the host where the Event Server role is configured to run, restore the Events Server directory from the Cloudera Manager 5 backup.

```
rm -rf /var/lib/cloudera-scm-eventserver/*
cp -rp /var/lib/cloudera-scm-eventserver_cm5cdh5/* /var/lib/cloudera-scm-eventserver/
```

3. Remove the Agent runtime state. Run the following command on all hosts:

```
rm -rf /var/run/cloudera-scm-agent /var/lib/cloudera-scm-agent/response.avro
```

4. On the host where the Service Monitor is running, restore the Service Monitor directory:

```
rm -rf /var/lib/cloudera-service-monitor/*
cp -rp /var/lib/cloudera-service-monitor_cm5cdh5/* /var/lib/cloudera-service-monitor/
```

5. On the host where the Host Monitor is running, restore the Host Monitor directory:

```
rm -rf /var/lib/cloudera-host-monitor/*
cp -rp /var/lib/cloudera-host-monitor_cm5cdh5/* /var/lib/cloudera-host-monitor/
```

6. Restore the Cloudera Navigator Solr storage directory from the CM5/CDH 5 backup.

```
rm -rf /var/lib/cloudera-scm-navigator/*
cp -rp /var/lib/cloudera-scm-navigator_cm5cdh5/* /var/lib/cloudera-scm-navigator/
```

7. On the Cloudera Manager Server, restore the `/etc/cloudera-scm-server/db.properties` file.

```
rm -rf /etc/cloudera-scm-server/db.properties
cp -rp cm5cdh5/etc/cloudera-scm-server/db.properties
/etc/cloudera-scm-server/db.properties
```

8. On each host in the cluster, restore the `/etc/cloudera-scm-agent/config.ini` file from your backup.

```
rm -rf /etc/cloudera-scm-agent/config.ini
cp -rp cm5cdh5/etc/cloudera-scm-agent/config.ini /etc/cloudera-scm-agent/config.ini
```

Start the Cloudera Manager Server and Agents

- Start the **Cloudera Manager Server**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo systemctl start cloudera-scm-server
```

If the Cloudera Manager Server starts without errors, no response displays.

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-server start
```

You should see the following:

```
Starting cloudera-scm-server: [ OK ]
```

- Hard Restart the **Cloudera Manager Agent**.

RHEL 7, SLES 12, Debian 8, Ubuntu 16.04 and higher

```
sudo /etc/init.d/cloudera-scm-agent next_stop_hard
sudo systemctl stop cloudera-scm-agent
```

RHEL 5 or 6, SLES 11, Debian 6 or 7, Ubuntu 12.04 or 14.04

```
sudo service cloudera-scm-agent hard_restart
```

- Start the **Cloudera Management Service**.

1. Log in to the Cloudera Manager Admin Console.

2. Select **Clusters** > **Cloudera Management Service**.
3. Select **Actions** > **Start**.

Upgrading Cloudera Navigator Data Encryption

[Cloudera Navigator Data Encryption](#) is a data-at-rest encryption and key management suite that includes [Cloudera Navigator Encrypt](#), [Cloudera Navigator Key Trustee Server](#), among other components. These can optionally be upgraded during a Cloudera Manager or CDH upgrade, and can also be upgraded individually.

Upgrading Cloudera Navigator Key Trustee Server

Navigator Key Trustee Server 5.4.x is the first release that supports installation using Cloudera Manager. If you are using Cloudera Manager, you must upgrade Key Trustee Server to 5.4 or higher using the command line or the [ktupgrade script](#) before you can migrate Key Trustee Server to Cloudera Manager control.

To upgrade Key Trustee Server from 3.8 to 5.5 or higher, use the [ktupgrade script](#) to simplify the upgrade process.

Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher

If you are upgrading Key Trustee Server from 3.8 to 5.5 or higher, see [Upgrading Cloudera Navigator Key Trustee Server 3.8 to 5.5 Using the ktupgrade Script](#) on page 163.



Note: Before upgrading Key Trustee Server, back up the Key Trustee Server. See [Backing Up and Restoring Key Trustee Server and Clients](#) for instructions.

Setting Up an Internal Repository

You must create an internal repository to install or upgrade the Cloudera Navigator data encryption components. For instructions on creating internal repositories (including Cloudera Manager, CDH, and Cloudera Navigator encryption components), see the following topics:

- [Configuring a Local Parcel Repository](#) on page 189
- [Configuring a Local Package Repository](#) on page 184

Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher Using Cloudera Manager

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)



Note: These instructions apply to using Cloudera Manager only. To upgrade Key Trustee Server using the command line, skip to the [Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher Using the Command Line \(CherryPy Web Server\)](#) on page 153 or [Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher Using the Command Line \(Apache Web Server\)](#) on page 154 section.

1. Add your internal parcel repository to Cloudera Manager following the instructions in [Configuring Cloudera Manager Server Parcel Settings](#).
2. Download, distribute, and activate the latest Key Trustee Server parcel on the cluster containing the Key Trustee Server host, following the instructions in [Managing Parcels](#).



Important: The KEYTRUSTEE parcel in Cloudera Manager is *not* the Key Trustee Server parcel; it is the Key Trustee KMS parcel. The parcel name for Key Trustee Server is KEYTRUSTEE_SERVER.

3. **(High Availability Key Trustee Servers Only)** Enable synchronous replication. On the active Key Trustee Server, run the following command:

```
sudo ktadmin enable-synchronous-replication --pg-rootdir /var/lib/keytrustee/db
```


Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher Using the Command Line (CherryPy Web Server)



Important: Use these instructions only if you have previously [migrated Key Trustee Server](#) to use the CherryPy web server instead of the Apache web server. Otherwise, skip to [Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher Using the Command Line \(Apache Web Server\)](#) on page 154.

The following instructions apply to both standalone and high availability Key Trustee Servers. For standalone Key Trustee Server, follow the instructions that refer to the *active* Key Trustee Server. For high availability Key Trustee Servers, follow the instructions on all Key Trustee Servers, unless otherwise indicated.

Upgrade Key Trustee Server

1. Stop the keytrusted service:

```
sudo service keytrusted stop
```

2. Install the EPEL Repository

Dependent packages are available through the Extra Packages for Enterprise Linux (EPEL) repository. To install the EPEL repository, install the `epel-release` package:

1. Copy the URL for the `epel-release-<version>.noarch` file for RHEL 6 or RHEL 7 located in the [How can I use these extra packages?](#) section of the EPEL wiki page.
2. Run the following commands to install the EPEL repository:

```
sudo wget <epel_rpm_url>
sudo yum install epel-release-<version>.noarch.rpm
```

Replace `<version>` with the version number of the downloaded RPM (for example, 6-8).

If the `epel-release` package is already installed, you see a message similar to the following:

```
Examining /var/tmp/yum-root-jmZhL0/epel-release-6-8.noarch.rpm: epel-release-6-8.noarch
/var/tmp/yum-root-jmZhL0/epel-release-6-8.noarch.rpm: does not update installed package.
Error: Nothing to do
```

Confirm that the EPEL repository is installed:

```
sudo yum repolist | grep -i epel
```

3. Install the Cloudera Repository

Add the internal repository you created. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.

Import the GPG key by running the following command:

```
sudo rpm --import http://repo.example.com/path/to/RPM-GPG-KEY-cloudera
```

4. Install the CDH Repository

Key Trustee Server and Key HSM depend on the `bigtop-utils` package, which is included in the CDH repository. For instructions on adding the CDH repository, see [Configuring a Local Package Repository](#) on page 184.

5. Upgrade Key Trustee Server:

```
sudo yum update keytrustee-server python-keytrustee
```

Upgrading Cloudera Navigator Data Encryption

6. Start the keytrusteed service:

```
sudo service keytrusteed start
```

(High Availability Key Trustee Servers Only) Enable Synchronous Replication

Run the following command on the active Key Trustee Server to enable synchronous replication after upgrading:

```
sudo ktadmin enable-synchronous-replication --pg-rootdir /var/lib/keytrustee/db
```

Migrate Key Trustee Server to Cloudera Manager

Skip to [Migrating Unmanaged Key Trustee Server to Cloudera Manager](#) on page 156 for instructions on migrating Key Trustee Server to Cloudera Manager control if you have not already done so during a previous upgrade.

Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher Using the Command Line (Apache Web Server)



Important: Use these instructions only if you have *not yet* [migrated Key Trustee Server](#) to use the CherryPy web server instead of the Apache web server. The Apache web server is not supported in versions 5.5 and higher.

The following instructions apply to both standalone and high availability Key Trustee Servers. For standalone Key Trustee Server, follow the instructions that refer to the *active* Key Trustee Server. For high availability Key Trustee Servers, follow the instructions on all Key Trustee Servers, unless otherwise indicated.

Migrate Apache Web Server to CherryPy



Note: Confirm that all ports listed in [Network Requirements](#) are open before proceeding.

For versions 5.4.0 and higher, Key Trustee Server uses CherryPy for the front end web interface; lower versions use the Apache web server. The Apache web server is not supported in versions 5.5 and higher. The CherryPy service is managed using the `keytrusteed` service. The Apache web server is managed using the `httpd` service. Before upgrading, run the following commands to migrate the web server from Apache to CherryPy.

1. On the active Key Trustee Server, run the `ktadmin db --configure` command as follows:

```
sudo ktadmin db --configure --port 11381 --pg-rootdir /var/lib/keytrustee/db --slave  
keytrustee02.example.com
```

Replace `keytrustee02.example.com` with the hostname of the passive Key Trustee Server. For standalone Key Trustee Server, omit the `--slave keytrustee02.example.com` portion of the command.

If you use a database directory other than `/var/lib/keytrustee/db`, create or edit the `/etc/sysconfig/keytrustee-db` file and add the following:

```
ARGS="--pg-rootdir /path/to/db"
```

2. Export the Key Trustee Server database. Run the following commands on the active Key Trustee Server:

```
sudo -u postgres pg_dump keytrustee > /var/lib/keytrustee/ktadbexport.pgsql  
chown keytrustee:keytrustee /var/lib/keytrustee/ktadbexport.pgsql
```

3. Start the Key Trustee Server database and import ktdbexport.pgsql:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db start --log
/var/lib/keytrustee/db/pg_ctl.log
sudo -u keytrustee /usr/pgsql-9.3/bin/createdb --host /tmp --port 11381 -O keytrustee
keytrustee
sudo -u keytrustee psql -d keytrustee -h /tmp -p 11381 <
/var/lib/keytrustee/ktdbexport.pgsql
```



Note: The `/etc/init.d/postgresql` script does not work when the PostgreSQL database is started by Key Trustee Server, and cannot be used to monitor the status of the database. Use `/etc/init.d/keytrustee-db` instead.

4. (High Availability Key Trustee Servers Only) Start the passive Key Trustee Server. Run the following commands on the passive Key Trustee Server:

```
sudo ktadmin --confdir /var/lib/keytrustee/.keytrustee init-slave --master
keytrustee01.example.com --pg-rootdir /var/lib/keytrustee/db --no-import-key
--master-host-port 11381 --logdir /var/lib/keytrustee/.keytrustee/logs
--postgres-config=local --no-start
```

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db start
```

If you use a database directory other than `/var/lib/keytrustee/db`, create or edit the `/etc/sysconfig/keytrustee-db` file and add the following:

```
ARGS="--pg-rootdir /path/to/db"
```

5. Edit `/var/lib/keytrustee/.keytrustee/keytrustee.conf` on all Key Trustee Servers to reference the new database and port. Set the `DB_CONNECT` parameter as follows:

```
"DB_CONNECT": "postgresql://localhost:11381/keytrustee?host=/tmp",
```

6. Restart the Apache web server. Run this command on all Key Trustee Servers:

```
sudo service httpd restart
```

7. Start the Key Trustee daemon (which starts the CherryPy web server). Run this command on all Key Trustee Servers:

```
sudo service keytrusteed start
```

8. After verifying that the Key Trustee daemon and CherryPy web server are running, stop the Apache web server and original database and prevent them from starting after reboots. Run these commands on all Key Trustee Servers:

```
sudo service httpd stop
sudo -u postgres /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/pgsql/9.3/keytrustee stop
sudo chkconfig httpd off
sudo chkconfig postgresql-9.3 off
```

Upgrade Key Trustee Server**1. Stop the httpd service:**

```
sudo service httpd stop
```

2. Install the EPEL Repository

Upgrading Cloudera Navigator Data Encryption

Dependent packages are available through the Extra Packages for Enterprise Linux (EPEL) repository. To install the EPEL repository, install the `epel-release` package:

1. Copy the URL for the `epel-release-<version>.noarch` file for RHEL 6 or RHEL 7 located in the [How can I use these extra packages?](#) section of the EPEL wiki page.
2. Run the following commands to install the EPEL repository:

```
sudo wget <epel_rpm_url>
sudo yum install epel-release-<version>.noarch.rpm
```

Replace `<version>` with the version number of the downloaded RPM (for example, 6-8).

If the `epel-release` package is already installed, you see a message similar to the following:

```
Examining /var/tmp/yum-root-jmZhL0/epel-release-6-8.noarch.rpm: epel-release-6-8.noarch
/var/tmp/yum-root-jmZhL0/epel-release-6-8.noarch.rpm: does not update installed package.
Error: Nothing to do
```

Confirm that the EPEL repository is installed:

```
sudo yum repolist | grep -i epel
```

3. Install the Cloudera Repository

Add the internal repository you created. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.

Import the GPG key by running the following command:

```
sudo rpm --import http://repo.example.com/path/to/RPM-GPG-KEY-cloudera
```

4. Upgrade Key Trustee Server:

```
sudo yum update keytrustee-server python-keytrustee
```

5. Start the httpd service:

```
sudo service httpd start
```

(High Availability Key Trustee Servers Only) Enable Synchronous Replication

Run the following command on the active Key Trustee Server to enable synchronous replication after upgrading:

```
sudo ktadmin enable-synchronous-replication --pg-rootdir /var/lib/keytrustee/db
```

Migrate Key Trustee Server to Cloudera Manager

Continue to [Migrating Unmanaged Key Trustee Server to Cloudera Manager](#) on page 156 for instructions on migrating Key Trustee Server to Cloudera Manager control if you have not already done so during a previous upgrade.

Migrating Unmanaged Key Trustee Server to Cloudera Manager

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

For simplified and centralized administration, perform the following steps to move Key Trustee Server under Cloudera Manager control (if you have not already done so) after upgrading Key Trustee Server:

1. **(Recommended)** Create a new cluster in Cloudera Manager containing only the hosts the Key Trustee Server will be installed on. Cloudera strongly recommends installing Key Trustee Server in a dedicated cluster to enable multiple clusters to share the same Key Trustee Server and to avoid restarting the Key Trustee Server when

restarting a cluster. See [Adding and Deleting Clusters](#) for instructions on how to create a new cluster in Cloudera Manager.

2. Download, distribute, and activate the Key Trustee Server parcel, following the instructions in [Managing Parcels](#). After you activate the Key Trustee Server parcel, Cloudera Manager prompts you to restart the cluster. Click the **Close** button to ignore this prompt. You *do not* need to restart the cluster after installing Key Trustee Server.
3. Stop the active and passive Key Trustee Server web servers using the command that corresponds to your backing web server. See [Migrate Apache Web Server to CherryPy](#) on page 154 for more information.

For Apache web servers:

```
sudo service httpd stop
```

For CherryPy web servers:

```
sudo service keytrusteed stop
```

4. Stop the active Key Trustee Server database. Run the following command on the active Key Trustee Server:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db stop
```



Warning: Do not stop the passive Key Trustee Server database. If it is stopped, start it before proceeding by running the following command on the passive Key Trustee Server:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db start
```

5. Add the Key Trustee Server service to your cluster, following the instructions in [Adding a Service](#). When customizing role assignments, assign the Active Key Trustee Server and Active Database roles to the active Key Trustee Server host, and the Passive Key Trustee Server and Passive Database roles to the passive Key Trustee Server host.
6. Stop the passive Key Trustee Server database. Run the following command on the passive Key Trustee Server:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db stop
```

7. Restart the Key Trustee Server service (**Key Trustee Server service** > **Actions** > **Restart**).



Important: Starting or restarting the Key Trustee Server service attempts to start the Active Database and Passive Database roles. If the Active Database is not running when the Passive Database attempts to start, the Passive Database fails to start. If this occurs, manually restart the Passive Database role after confirming that the Active Database role is running.

8. **(High Availability Key Trustee Servers Only)** Enable synchronous replication. Run the following command on the active Key Trustee Server:

```
sudo ktadmin enable-synchronous-replication --pg-rootdir /var/lib/keytrustee/db
```

Upgrading Cloudera Navigator Key Trustee Server 3.x to 5.4.x

Navigator Key Trustee Server 5.4.x is the first release that supports installation using Cloudera Manager. If you are using Cloudera Manager, then you must upgrade Key Trustee Server using the command line before you can migrate Key Trustee Server to Cloudera Manager control.

To upgrade Key Trustee Server to 5.5 or higher, see [Upgrading Cloudera Navigator Key Trustee Server 3.8 to 5.5 Using the ktupgrade Script](#) on page 163.



Note: Before upgrading Key Trustee Server, back up the Key Trustee Server database and configuration directory. See [Backing Up Key Trustee Server Manually](#) for instructions.

Upgrading Key Trustee Server 3.x to 5.4.x Using the Command Line

The following instructions apply to both standalone and high availability Key Trustee Servers. For standalone Key Trustee Server, follow the instructions that refer to the *active* Key Trustee Server. For high availability Key Trustee Servers, follow the instructions on all Key Trustee Servers, unless otherwise indicated.

Upgrade Key Trustee Server

1. Stop the httpd service:

```
sudo service httpd stop
```

2. Install the EPEL Repository

Dependent packages are available through the Extra Packages for Enterprise Linux (EPEL) repository. To install the EPEL repository, install the `epel-release` package:

1. Copy the URL for the `epel-release-<version>.noarch` file for RHEL 6 or RHEL 7 located in the [How can I use these extra packages?](#) section of the EPEL wiki page.
2. Run the following commands to install the EPEL repository:

```
sudo wget <epel_rpm_url>  
sudo yum install epel-release-<version>.noarch.rpm
```

Replace `<version>` with the version number of the downloaded RPM (for example, 6-8).

If the `epel-release` package is already installed, you see a message similar to the following:

```
Examining /var/tmp/yum-root-jmZhL0/epel-release-6-8.noarch.rpm: epel-release-6-8.noarch  
/var/tmp/yum-root-jmZhL0/epel-release-6-8.noarch.rpm: does not update installed package.  
Error: Nothing to do
```

Confirm that the EPEL repository is installed:

```
sudo yum repolist | grep -i epel
```

3. Install the Cloudera Repository

Create or edit the `/etc/yum.repos.d/gazzang.repo` file (for example, `sudo vi /etc/yum.repos.d/gazzang.repo`) and add the following text. Replace `USER` and `PASSWD` with the username and password provided by Cloudera. If you do not know your username or password, contact your Cloudera account team.

```
[gazzang_stable]  
name=RHEL $releasever - gazzang.com - base  
baseurl=https://USER:PASSWD@archive.gazzang.com/redhat/stable/$releasever  
enabled=1  
gpgcheck=1  
gpgkey=http://archive.gazzang.com/gpg_gazzang.asc
```



Important: If you are using CentOS, add the following line to the CentOS base repository:

```
exclude=python-psycopg2*
```

By default, the base repository is located at `/etc/yum.repos.d/CentOS-Base.repo`. If you have an internal mirror of the base repository, update the correct file for your environment.

Import the GPG key by running the following command:

```
sudo rpm --import http://archive.gazzang.com/gpg_gazzang.asc
```

4. Upgrade Key Trustee Server using yum:

```
sudo yum update keytrustee-server python-keytrustee
```

5. Start the httpd service:

```
sudo service httpd start
```

Migrate Apache Web Server to CherryPy



Note: Confirm that all ports listed in [Network Requirements](#) are open before proceeding.

For versions 5.4.0 and higher, Key Trustee Server uses CherryPy for the front end web interface; lower versions use the Apache web server. The CherryPy service is managed using the `keytrustee` service. The Apache web server is managed using the `httpd` service. Run the following commands to migrate the web server from Apache to CherryPy.

1. On the active Key Trustee Server, run the `ktadmin db --configure` command as follows:

```
sudo -u keytrustee ktadmin db --configure --port 11381 --pg-rootdir /var/lib/keytrustee/db
--slave keytrustee02.example.com
```

Replace `keytrustee02.example.com` with the hostname of the passive Key Trustee Server. For standalone Key Trustee Server, omit the `--slave keytrustee02.example.com` portion of the command.

2. Export the active Key Trustee Server database. Run the following commands on the active Key Trustee Server:

```
sudo -u postgres pg_dump keytrustee > /var/lib/keytrustee/ktdbexport.pgsql
chown keytrustee:keytrustee /var/lib/keytrustee/ktdbexport.pgsql
```

3. Start the Key Trustee Server database and import `ktdbexport.pgsql`:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db start --log
/var/lib/keytrustee/db/pg_ctl.log
sudo -u keytrustee /usr/pgsql-9.3/bin/createdb --host /tmp --port 11381 -O keytrustee
keytrustee
sudo -u keytrustee psql -d keytrustee -h /tmp -p 11381 <
/var/lib/keytrustee/ktdbexport.pgsql
```



Note: The `/etc/init.d/postgresql` script does not work when the PostgreSQL database is started by Key Trustee Server, and cannot be used to monitor the status of the database. Use `/etc/init.d/keytrustee-db` instead.

- 4. (High Availability Key Trustee Servers Only)** Start the passive Key Trustee Server. Run the following commands on the passive Key Trustee Server:

```
sudo -u keytrustee ktadmin --confdir /var/lib/keytrustee/.keytrustee init-slave --master
keytrustee01.example.com --pg-rootdir /var/lib/keytrustee/db --no-import-key
--master-host-port 11381 --logdir /var/lib/keytrustee/.keytrustee/logs
--postgres-config=local --no-start
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db start
```

- 5.** Edit `/var/lib/keytrustee/.keytrustee/keytrustee.conf` on the active and passive Key Trustee Servers to reference the new database and port. Set the `DB_CONNECT` parameter as follows:

```
"DB_CONNECT": "postgresql://localhost:11381/keytrustee?host=/tmp",
```

- 6.** Restart the Apache web server. Run this command on all Key Trustee Servers:

```
sudo service httpd restart
```

- 7.** Start the Key Trustee daemon (which starts the CherryPy web server). Run this command on all Key Trustee Servers:

```
sudo /etc/init.d/keytrusteed start
```

- 8.** After verifying that the Key Trustee daemon and CherryPy web server are running, stop the Apache web server and original database and prevent them from starting after reboots. Run these commands on all Key Trustee Servers:

```
sudo service httpd stop
sudo -u postgres /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/pgsql/9.3/keytrustee stop
sudo chkconfig httpd off
sudo chkconfig postgresql-9.3 off
```

(High Availability Key Trustee Servers Only) Enable Synchronous Replication

Run the following command on the active Key Trustee Server to enable synchronous replication after upgrading:

```
sudo -u keytrustee ktadmin enable-synchronous-replication --pg-rootdir
/var/lib/keytrustee/db
```

Migrating Unmanaged Key Trustee Server to Cloudera Manager



Important: If you are upgrading to Key Trustee Server 5.5 or higher without the [ktupgrade script](#), skip this step and continue to [Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher](#) on page 152.

For simplified and centralized administration, perform the following steps to move Key Trustee Server under Cloudera Manager control after upgrading Key Trustee Server:

- 1.** Download the Key Trustee Server CSD from the following location:

```
https://archive.gazzang.com/parcels/cloudera/keytrustee-server/5.4.9/stable/latest/csd/
```

When prompted, enter your credentials. If you do not know your credentials, contact your Cloudera account team.

- 2.** Install the CSD into Cloudera Manager as instructed in [Custom Service Descriptor Files](#). The CSD can only be installed on parcel-deployed clusters.

3. Add the following parcel repository to Cloudera Manager following the instructions in [Configuring Cloudera Manager Server Parcel Settings](#):

```
https://<username>:<password>@archive.gazsang.com/parcels/cloudera/keytrustee-server/5.4.9/stable/latest
```

Replace `<username>` and `<password>` with your credentials. If you do not know your credentials, contact your Cloudera account team.

4. **(Recommended)** Create a new cluster in Cloudera Manager containing only the hosts the Key Trustee Server will be installed on. Cloudera strongly recommends installing Key Trustee Server in a dedicated cluster to enable multiple clusters to share the same Key Trustee Server and to avoid restarting the Key Trustee Server when restarting a cluster. See [Adding and Deleting Clusters](#) for instructions on how to create a new cluster in Cloudera Manager.
5. Download, distribute, and activate the Key Trustee Server parcel, following the instructions in [Managing Parcels](#). After you activate the Key Trustee Server parcel, Cloudera Manager prompts you to restart the cluster. Click the **Close** button to ignore this prompt. You *do not* need to restart the cluster after installing Key Trustee Server.
6. Stop the active and passive Key Trustee Server web servers using the command that corresponds to your backing web server. See [Migrate Apache Web Server to CherryPy](#) on page 159 for more information.

For Apache web servers:

```
sudo service httpd stop
```

For CherryPy web servers:

```
sudo service keytrusteed stop
```

7. Stop the active Key Trustee Server database. Run the following command on the active Key Trustee Server:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db stop
```



Warning: Do not stop the passive Key Trustee Server database. If it is stopped, start it before proceeding by running the following command on the passive Key Trustee Server:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db start
```

8. Add the Key Trustee Server service to your cluster, following the instructions in [Adding a Service](#). When customizing role assignments, assign the Active Key Trustee Server and Active Database roles to the active Key Trustee Server host, and the Passive Key Trustee Server and Passive Database roles to the passive Key Trustee Server host.
9. Stop the passive Key Trustee Server database. Run the following command on the passive Key Trustee Server:

```
sudo -u keytrustee /usr/pgsql-9.3/bin/pg_ctl -D /var/lib/keytrustee/db stop
```

- 10 Restart the Key Trustee Server service (**Key Trustee Server service > Actions > Restart**).



Important: Starting or restarting the Key Trustee Server service attempts to start the Active Database and Passive Database roles. If the Active Database is not running when the Passive Database attempts to start, the Passive Database fails to start. If this occurs, manually restart the Passive Database role after confirming that the Active Database role is running.

- 11. (High Availability Key Trustee Servers Only)** Enable synchronous replication. Run the following command on the active Key Trustee Server:

```
sudo -u keytrustee ktadmin enable-synchronous-replication --pg-rootdir /var/lib/keytrustee/db
```

Updating Key Trustee Server Clients

After upgrading Key Trustee Server to 5.4 or higher, you must configure Key Trustee Server clients (namely Key Trustee KMS and Cloudera Navigator Encrypt) to communicate with Key Trustee Server over the new ports:

- **Key Trustee KMS**

Add the following entries to the Key Trustee KMS advanced configuration snippet (**Key Trustee KMS service > Configuration > Advanced > Key Management Server Advanced Configuration Snippet (Safety Valve) for kms-site.xml**):

```
<property>
  <name>cloudera.trustee.keyprovider.hkpport</name>
  <value>hkp_port_number</value>
  <description>
    Indicates the HTTP port on which Key Trustee Server clients should request public
    keys.
    On Key Trustee Server 3.8 (Apache webserver-based) servers, this is usually port
    80 (unencrypted).
    On Key Trustee Server 5.4 and higher (CherryPy-based) servers, this is usually
    port 11371 (SSL-encrypted).
  </description>
</property>
<property>
  <name>cloudera.trustee.keyprovider.ktsport</name>
  <value>kts_port_number</value>
  <description>
    Indicates the HTTPS port on which the client sends and receives Key Trustee Server
    protocol messages.
    On Key Trustee Server 3.8 (Apache webserver-based) servers, this is usually port
    443 (SSL-encrypted).
    On Key Trustee Server 5.4 and higher (CherryPy-based) servers, this is usually
    port 11371 (SSL-encrypted).
  </description>
</property>
<property>
  <name>cloudera.trustee.keyprovider.hkpssl</name>
  <value>boolean</value>
  <description>
    Indicates whether the client should communicate with the HKP server over an
    SSL-encrypted (true) or unencrypted (false) channel.
    On Key Trustee Server 3.8 (Apache webserver-based) servers, this is usually false
    (unencrypted).
    On Key Trustee Server 5.4 and higher (CherryPy-based) servers, this is usually
    true (SSL-encrypted).
  </description>
</property>
```

- **Cloudera Navigator Encrypt**

See [Updating Key Trustee Server Ports](#) for instructions on updating Cloudera Navigator Encrypt to use the new ports.

Validating Key Operations

Verify that the upgrade was successful by running the following command on all Key Trustee Servers. The output should be similar to the following. If high availability is enabled, the output should be identical on all Key Trustee Servers:

```
curl -k https://keytrustee.example.com:11371/?a=fingerprint
4096R/4EDC46882386C827E20DEEA2D850ACA33BEDB0D1
```

Replace `keytrustee.example.com` with the fully qualified domain name (FQDN) of each Key Trustee Server you are validating.

If you are using Key Trustee Server as the backing key store for [HDFS Transparent Encryption](#), run the following commands to verify that Hadoop key operations are successful:

```
hadoop key create hadoop_test_key
hadoop key list
hadoop key delete hadoop_test_key
```

Upgrading Cloudera Navigator Key Trustee Server 3.8 to 5.5 Using the ktupgrade Script

Cloudera provides a Python script (`ktupgrade`) to simplify upgrading Key Trustee Server 3.8 to 5.5. The script upgrades package-based Key Trustee Server 3.8 to package-based Key Trustee Server 5.5 and switches the web server from Apache to CherryPy. After the upgrade completes, you must manually migrate Key Trustee Server to use parcels and be managed by Cloudera Manager.

To upgrade from 3.x to 5.5 manually, you must first upgrade to 5.4, and then upgrade to 5.5:

- [Upgrading Cloudera Navigator Key Trustee Server 3.x to 5.4.x](#) on page 157
- [Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher](#) on page 152

Prerequisites



Important: The `ktupgrade` script supports upgrading from version 3.8.0 or 3.8.1 to version 5.5.0 or 5.5.2 only. To upgrade to a version higher than 5.5.2, use the `ktupgrade` script to upgrade to 5.5.2, and then follow the instructions in [Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher](#) on page 152 to upgrade to the version you want.

- Before upgrading Key Trustee Server, upgrade Cloudera Manager and CDH. See [Upgrading Cloudera Manager](#) on page 38 and [Upgrading the CDH Cluster](#) on page 92. If you are upgrading Key Trustee Server to a version higher than 5.5.2, you can upgrade Cloudera Manager and CDH directly to the version you want before continuing; you do not need to upgrade Cloudera Manager and CDH to 5.5 and complete the Key Trustee Server upgrade before upgrading Cloudera Manager and CDH to a higher version. The Cloudera Manager version must be equal to or higher than the Key Trustee Server version. See [Product Compatibility Matrix for Cloudera Navigator Encryption](#) for more information.
- If you are using [HDFS Transparent Encryption](#) with Key Trustee Server, upgrade Key Trustee KMS. See [Upgrading Key Trustee KMS](#) on page 170 for instructions.
- You must run the `ktupgrade` script as `root`.
- The `ktupgrade` script uses `yum` to upgrade Key Trustee Server. If the Key Trustee Server host does not have Internet access, you must download the Key Trustee Server dependencies from a host with Internet access and copy them to the Key Trustee Server host:

1. Create a temporary directory to store the packages:

```
mkdir tmp-keytrustee
```

2. Download the `bigtop-utils` package from the CDH repository:

```
sudo wget -P tmp-keytrustee <url>
```

Replace `<url>` with the URL corresponding to the Key Trustee Server version to which you are upgrading:

Table 3: URL for bigtop-utils Package

Key Trustee Server Version	URL
5.5.2	https://archive.cloudera.com/cdh5/redhat/6/x86_64/cdh/5.5.2/RPMS/noarch/bigtop-utils-0.7.0+cdh5.5.2+0-1.cdh5.5.2.p0.10.el6.noarch.rpm
5.5.0	https://archive.cloudera.com/cdh5/redhat/6/x86_64/cdh/5.5.0/RPMS/noarch/bigtop-utils-0.7.0+cdh5.5.0+0-1.cdh5.5.0.p0.15.el6.noarch.rpm

3. Download the python-paste and python-cherrypy packages:

```
sudo yum install yum-downloadonly
sudo yum install --downloadonly --downloadaddir=tmp-keytrustee/ python-paste python-cherrypy
```

4. Copy the packages to the Key Trustee Server host:

```
sudo scp tmp-keytrustee/*.rpm <username>@kts01.example.com:/path/to/tmp-keytrustee
```

Replace *kts01.example.com* with the hostname of the active Key Trustee Server, and */path/to/tmp-keytrustee* with the path to a directory to which you have access.

Download the ktupgrade Script and Repository Tarball

1. Download the ktupgrade script on the active Key Trustee Server host:

```
sudo wget http://archive.gazzang.com/keytrustee/ktupgrade
```

If the Key Trustee Server host does not have Internet access, run the command on an Internet-connected host, and then copy the file to the active Key Trustee Server host.

2. Download the repository tarball for Key Trustee Server [5.5.0](#) or [5.5.2](#):

- Select **Packages** from the **SELECT DOWNLOAD TYPE** drop-down menu.
- Select your operating system from the **SELECT AN OS** drop-down menu.
- Click **DOWNLOAD NOW**.
- Copy the downloaded file to the active Key Trustee Server host. Make sure you put the repository tarball and *ktupgrade* script in the same directory.

Run the ktupgrade Script



Important: You must run the *ktupgrade* script as the *root* user. By default, the script upgrades the active Key Trustee Server, and then connects to the passive Key Trustee Server host as *root* over SSH (if you are using Key Trustee Server high availability) to upgrade it. You are prompted twice for the *root* password (first to copy the files, and then for the SSH connection).

If your environment does not allow *root* to log in over SSH, contact [Cloudera Support](#) for assistance.

Upgrade the Active Key Trustee Server Using the ktupgrade Script

1. On the active Key Trustee Server host, change to the directory that contains the ktupgrade script and the repository tarball:

```
# cd /path/to/tmp-keytrustee
```

If the host does not have Internet access, make sure that the dependency files you downloaded in [Prerequisites](#) on page 163 are in the same directory as the script and tarball.

2. Make sure the script is executable:

```
# chmod a+x ktupgrade
```

3. Run the ktupgrade script as follows:

```
# ./ktupgrade upgrade-active-kts key-trustee-server-5.5.2-el6.tar.gz
```

Replace *key-trustee-server-5.5.2-el6.tar.gz* with the file name of the repository tarball you downloaded in [Download the ktupgrade Script and Repository Tarball](#) on page 164.

Downgrade Key Trustee Server Using the ktupgrade Script

If you experience any problems upgrading Key Trustee Server, you can use the script to downgrade to your previous version. Run the following command on the active Key Trustee Server:

```
# cd /path/to/tmp-keytrustee
# ./ktupgrade downgrade-active-kts
```

Migrate Key Trustee Server to Cloudera Manager

Minimum Required Role: [Cluster Administrator](#) (also provided by **Full Administrator**)

Before continuing, you must create an internal repository for the Key Trustee Server parcel. For instructions on creating internal repositories (including Cloudera Manager, CDH, and Cloudera Navigator encryption components), see [Configuring a Local Parcel Repository](#) on page 189.

After creating the internal Key Trustee Server parcel repository, do the following:

1. Create a new cluster in Cloudera Manager containing only the Key Trustee Server hosts. This enables multiple clusters to share the same Key Trustee Server and avoids restarting Key Trustee Server when restarting a cluster. See [Adding and Deleting Clusters](#) for instructions on how to create a new cluster in Cloudera Manager.
2. Download, distribute, and activate the Key Trustee Server parcel, following the instructions in [Managing Parcels](#). After you activate the Key Trustee Server parcel, Cloudera Manager prompts you to restart the cluster. Click the **Close** button to ignore this prompt. You *do not* need to restart the cluster after installing Key Trustee Server.
3. Stop the active and passive Key Trustee Server web servers by running the following command on all Key Trustee Server hosts:

```
sudo -u keytrustee service keytrusteed stop
```

4. Stop the active Key Trustee Server database by running the following command on the active Key Trustee Server:

```
sudo -u keytrustee service keytrustee-db stop
```



Warning: Do not stop the passive Key Trustee Server database. If it is stopped, start it before proceeding by running the following command on the passive Key Trustee Server:

```
sudo -u keytrustee service keytrustee-db start
```

5. Add the Key Trustee Server service to your cluster, following the instructions in [Adding a Service](#). When customizing role assignments, assign the Active Key Trustee Server and Active Database roles to the active Key Trustee Server host, and the Passive Key Trustee Server and Passive Database roles to the passive Key Trustee Server host.
6. Stop the passive Key Trustee Server database. Run the following command on the passive Key Trustee Server:

```
sudo -u keytrustee service keytrustee-db stop
```

7. Restart the Key Trustee Server service (**Key Trustee Server service** > **Actions** > **Restart**).



Important: Starting or restarting the Key Trustee Server service attempts to start the Active Database and Passive Database roles. If the Active Database is not running when the Passive Database attempts to start, the Passive Database fails to start. If this occurs, manually restart the Passive Database role after confirming that the Active Database role is running.

8. (High Availability Key Trustee Servers Only) Enable synchronous replication. Run the following command on the active Key Trustee Server:

```
sudo -u keytrustee ktadmin enable-synchronous-replication --pg-rootdir /var/lib/keytrustee/db
```

Validate Key Operations

Verify that the upgrade was successful by running the following command on all Key Trustee Servers. The output should be similar to the following. If high availability is enabled, the output should be identical on all Key Trustee Servers:

```
$ curl -k https://keytrustee.example.com:11371/?a=fingerprint
4096R/4EDC46882386C827E20DEEA2D850ACA33BEDB0D1
```

Replace `keytrustee.example.com` with the fully qualified domain name (FQDN) of each Key Trustee Server you are validating.

If you are using Key Trustee Server as the backing key store for [HDFS Transparent Encryption](#), run the following commands to verify that Hadoop key operations are successful:

```
hadoop key create hadoop_test_key
hadoop key list
hadoop key delete hadoop_test_key
```

Updating Key Trustee Server Clients

After upgrading Key Trustee Server to 5.4 or higher, you must configure Key Trustee Server clients (namely Key Trustee KMS and Cloudera Navigator Encrypt) to communicate with Key Trustee Server over the new ports:

- **Key Trustee KMS**

Add the following entries to the Key Trustee KMS advanced configuration snippet (**Key Trustee KMS service > Configuration > Advanced > Key Management Server Advanced Configuration Snippet (Safety Valve) for kms-site.xml**):

```
<property>
  <name>cloudera.trustee.keyprovider.hkpport</name>
  <value>hkp_port_number</value>
  <description>
    Indicates the HTTP port on which Key Trustee Server clients should request public
    keys.
    On Key Trustee Server 3.8 (Apache webserver-based) servers, this is usually port
    80 (unencrypted).
    On Key Trustee Server 5.4 and higher (CherryPy-based) servers, this is usually
    port 11371 (SSL-encrypted).
  </description>
</property>
<property>
  <name>cloudera.trustee.keyprovider.ktsport</name>
  <value>kts_port_number</value>
  <description>
    Indicates the HTTPS port on which the client sends and receives Key Trustee Server
    protocol messages.
    On Key Trustee Server 3.8 (Apache webserver-based) servers, this is usually port
    443 (SSL-encrypted).
    On Key Trustee Server 5.4 and higher (CherryPy-based) servers, this is usually
    port 11371 (SSL-encrypted).
  </description>
</property>
```

```

</property>
<property>
  <name>cloudera.trustee.keyprovider.hkpssl</name>
  <value>boolean</value>
  <description>
    Indicates whether the client should communicate with the HKP server over an
    SSL-encrypted (true) or unencrypted (false) channel.
    On Key Trustee Server 3.8 (Apache webserver-based) servers, this is usually false
    (unencrypted).
    On Key Trustee Server 5.4 and higher (CherryPy-based) servers, this is usually
    true (SSL-encrypted).
  </description>
</property>

```

- **Cloudera Navigator Encrypt**

See [Updating Key Trustee Server Ports](#) for instructions on updating Cloudera Navigator Encrypt to use the new ports.

Validating Key Operations

Verify that the upgrade was successful by running the following command on all Key Trustee Servers. The output should be similar to the following. If high availability is enabled, the output should be identical on all Key Trustee Servers:

```
curl -k https://keytrustee.example.com:11371/?a=fingerprint
4096R/4EDC46882386C827E20DEEA2D850ACA33BEDB0D1
```

Replace `keytrustee.example.com` with the fully qualified domain name (FQDN) of each Key Trustee Server you are validating.

If you are using Key Trustee Server as the backing key store for [HDFS Transparent Encryption](#), run the following commands to verify that Hadoop key operations are successful:

```
hadoop key create hadoop_test_key
hadoop key list
hadoop key delete hadoop_test_key
```

(Optional) Upgrade to a Higher Release

If you are upgrading Key Trustee Server to a version higher than 5.5.2, continue to [Upgrading Cloudera Navigator Key Trustee Server 5.4.x or Higher](#) on page 152.

Upgrading Cloudera Navigator Key HSM

Setting Up an Internal Repository

You must create an internal repository to install or upgrade Cloudera Navigator Key HSM. For instructions on creating internal repositories (including Cloudera Manager, CDH, and Cloudera Navigator encryption components), see [Configuring a Local Package Repository](#) on page 184.

Upgrading Key HSM (Minor and Patch Version Upgrades)

If you are upgrading from Key HSM 1.x (shipped with CDH 5.x and earlier) to Key HSM 6.x, use the instructions in [Upgrading Key HSM \(Major Version Upgrades\)](#) on page 168; do *not* use the procedure documented in this section.



Important: If you have implemented Key Trustee Server high availability, upgrade Key HSM on each Key Trustee Server.

1. Install the Cloudera Repository

Upgrading Cloudera Navigator Data Encryption

Add the internal repository you created. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.

Import the GPG key by running the following command:

```
sudo rpm --import http://repo.example.com/path/to/RPM-GPG-KEY-cloudera
```

2. Install the CDH Repository

Key Trustee Server and Key HSM depend on the `bigtop-utils` package, which is included in the CDH repository. For instructions on adding the CDH repository, see [Configuring a Local Package Repository](#) on page 184.

3. Stop the Key HSM Service

Stop the Key HSM service before upgrading:

```
sudo service keyhsm shutdown
```

4. Upgrade Navigator Key HSM

Upgrade the Navigator Key HSM package using `yum`:

```
sudo yum update keytrustee-keyhsm
```

Cloudera Navigator Key HSM is installed to the `/usr/share/keytrustee-server-keyhsm` directory by default.

5. Start the Key HSM Service

Start the Key HSM service:

```
sudo service keyhsm start
```

Upgrading Key HSM (Major Version Upgrades)



Important: Only use this procedure if you are upgrading from Key HSM 1.x (shipped with CDH 5.x and earlier) to Key HSM 6.x. There is a unique configuration issue that impacts this upgrade scenario, and the steps here are different from those required for all minor Key HSM upgrades. This procedure is *not* applicable to minor version or patch release upgrades.

1. Install the Cloudera Repository

Add the internal repository you created. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.

Import the GPG key by running the following command:

```
sudo rpm --import http://repo.example.com/path/to/RPM-GPG-KEY-cloudera
```

2. Install the CDH Repository

Key Trustee Server and Key HSM depend on the `bigtop-utils` package, which is included in the CDH repository. For instructions on adding the CDH repository, see [Configuring a Local Package Repository](#) on page 184.

3. Stop the Key HSM Service

Stop the Key HSM service before upgrading:

```
sudo service keyhsm shutdown
```

4. Upgrade Navigator Key HSM

Upgrade the Navigator Key HSM package using `yum`:

```
sudo yum update keytrustee-keyhsm
```

Cloudera Navigator Key HSM is installed to the `/usr/share/keytrustee-server-keyhsm` directory by default.

5. Rename Previously-Created Configuration Files

For Key HSM major version upgrades, previously-created configuration files do not authenticate with the HSM and Key Trustee Server, so you must recreate these files by re-executing the `setup` and `trust` commands. First, navigate to the Key HSM installation directory and rename the `application.properties`, `keystore`, and `truststore` files:

```
cd /usr/share/keytrustee-server-keyhsm/
mv application.properties application.properties.bak
mv keystore keystore.bak
mv truststore truststore.bak
```

6. Initialize Key HSM

Run the `service keyhsm setup` command in conjunction with the name of the target HSM distribution:

```
sudo service keyhsm setup [keysecure|thales|luna]
```

For more details, see [Initializing Navigator Key HSM](#).

7. Establish Trust Between Key HSM and the Key Trustee Server

The Key HSM service must explicitly trust the Key Trustee Server certificate (presented during TLS handshake). To establish this trust, run the following command:

```
sudo keyhsm trust /path/to/key_trustee_server/cert
```

For more details, see [Establish Trust from Key HSM to Key Trustee Server](#).

8. Start the Key HSM Service

Start the Key HSM service:

```
sudo service keyhsm start
```

9. Establish Trust Between Key Trustee Server and Key HSM

Establish trust between the Key Trustee Server and the Key HSM by specifying the path to the private key and certificate:

```
sudo ktadmin keyhsm --server https://keyhsm01.example.com:9090 \
--client-certfile /etc/pki/cloudera/certs/mycert.crt \
--client-keyfile /etc/pki/cloudera/certs/mykey.key --trust
```

For a password-protected Key Trustee Server private key, add the `--passphrase` argument to the command (enter the password when prompted):

```
sudo ktadmin keyhsm --passphrase \
--server https://keyhsm01.example.com:9090 \
--client-certfile /etc/pki/cloudera/certs/mycert.crt \
--client-keyfile /etc/pki/cloudera/certs/mykey.key --trust
```

For additional details, see [Integrate Key HSM and Key Trustee Server](#).

10 Remove Configuration Files From Previous Installation

Upgrading Cloudera Navigator Data Encryption

After completing the upgrade, remove the saved configuration files from the previous installation:

```
cd /usr/share/keytrustee-server-keyhsm/  
rm application.properties.bak  
rm keystore.bak  
rm truststore.bak
```

Upgrading Key Trustee KMS



Important: Following these instructions upgrades the software for the Key Trustee KMS service; this enables you to use Cloudera Navigator Key Trustee Server as the underlying keystore for [HDFS Transparent Encryption](#). This *does not* upgrade Key Trustee Server. See [Upgrading Cloudera Navigator Key Trustee Server](#) on page 152 for instructions on upgrading Key Trustee Server.

Key Trustee KMS is supported only in Cloudera Manager deployments. You can install the software using parcels or packages, but running Key Trustee KMS outside of Cloudera Manager is not supported.

Setting Up an Internal Repository

You must create an internal repository to upgrade Key Trustee KMS. For instructions on creating internal repositories (including Cloudera Manager, CDH, and Cloudera Navigator encryption components), see [Configuring a Local Parcel Repository](#) on page 189 if you are using parcels, or [Configuring a Local Package Repository](#) on page 184 if you are using packages.

Validating Private Key Synchronization (Key Trustee KMS HA Only)

Key Trustee KMS provides logic to detect and warn users about a potential problem where the GPG private keys have not been properly synchronized across all Key Trustee KMS HA hosts. If you have been running Key Trustee KMS on different hosts, and have not maintained private key synchronization, it is possible that the hosts may continue to operate and appear in a healthy state. However, when private keys are not synchronized between hosts, you can end up in a "split brain" scenario. In this scenario, the keys are actually only intermittently accessible, depending on which Key Trustee KMS host a client interacts with, because cryptographic key material encrypted by one Key Trustee KMS host cannot be decrypted by another. In the event of a catastrophic failure of one Key Trustee KMS host, any keys it has encrypted and any data encrypted by those keys will become inaccessible.

Key Trustee KMS detects this error state using a GPG validation check, which runs automatically when the Key Trustee KMS is restarted as part of the upgrade process. When the validation check discovers that private keys between hosts do not match, it returns the following error and aborts the restart operation:

```
java.io.IOException: Unable to verify private key match between KMS hosts. Verify private  
key files have been synced  
between all KMS hosts. Aborting to prevent data inconsistency.
```

To determine whether or not the Key Trustee KMS private keys are different, compare the MD5 hash of the private keys by executing the following command on each Key Trustee KMS host:

```
md5sum /var/lib/kms-keytrustee/keytrustee/.keytrustee/secring.gpg
```



Warning: If you see the previously mentioned Java exception in the Key Trustee KMS logs after an upgrade, or independently verify that the GPG private keys do not match using the MD5 hash, do *not* attempt to synchronize existing keys. If you overwrite the private key and do not have a backup, any keys encrypted by that private key are permanently inaccessible, and any data encrypted by those keys is permanently irretrievable. If the private keys are preserved, the problem is recoverable and no data should be lost. Immediately back up all Key Trustee KMS hosts, and contact Cloudera Support for assistance correcting the issue.

Upgrading Key Trustee KMS Using Parcels



Important: Back up Key Trustee KMS before upgrading. See [Backing Up and Restoring Key Trustee Server and Clients](#) for instructions.

1. Go to **Hosts > Parcels**.
2. Click **Configuration** and add your internal repository to the **Remote Parcel Repository URLs** section. See [Configuring Cloudera Manager to Use an Internal Remote Parcel Repository](#) on page 192 for more information.
3. Click **Save Changes**.
4. Download, distribute, and activate the KEYTRUSTEE parcel for the version to which you are upgrading. See [Parcels](#) for detailed instructions on using parcels to install or upgrade components.
5. Restart the Key Trustee KMS service (**Key Trustee KMS service > Actions > Restart**).

Upgrading Key Trustee KMS Using Packages

1. After [Setting Up an Internal Repository](#) on page 170, configure the Key Trustee KMS host to use the repository. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.
2. Add the CDH repository. See [Configuring a Local Package Repository](#) on page 184 for instructions.
3. Upgrade the keytrustee-keyprovider package using the appropriate command for your operating system:

- **RHEL-compatible**

```
sudo yum install keytrustee-keyprovider
```

- **SLES**

```
sudo zypper install keytrustee-keyprovider
```

- **Ubuntu or Debian**

```
sudo apt-get install keytrustee-keyprovider
```

4. Restart the Key Trustee KMS service (**Key Trustee KMS service > Actions > Restart**).

Upgrading Cloudera Navigator HSM KMS



Important: HSM KMS is supported only in Cloudera Manager deployments. You can install the software using parcels or packages, but running HSM KMS outside of Cloudera Manager is not supported.

Setting Up an Internal Repository

You must create an internal repository to upgrade HSM KMS. For instructions on creating internal repositories (including Cloudera Manager, CDH, and Cloudera Navigator encryption components), see [Configuring a Local Parcel Repository](#) on page 189 if you are using parcels, or [Configuring a Local Package Repository](#) on page 184 if you are using packages.

Upgrading HSM KMS Using Parcels

For customers using the Thales HSM KMS, do not use these upgrade steps. Instead, use [Upgrading 5.x to 6.x Parcels for Thales HSM KMS](#) on page 172.

To upgrade an HSM KMS using parcels:

1. Go to **Hosts > Parcels**.

2. Click **Configuration** and add your internal repository to the **Remote Parcel Repository URLs** section. See [Configuring Cloudera Manager to Use an Internal Remote Parcel Repository](#) on page 192 for more information.
3. Click **Save Changes**.
4. Download, distribute, and activate the KEYTRUSTEE parcel for the version to which you are upgrading. See [Parcels](#) for detailed instructions on using parcels to install or upgrade components.
5. Restart the HSM KMS service (**HSM KMS service > Actions > Restart**).



Note: After you activate the KEYTRUSTEE parcel, Cloudera Manager displays the **Restart** button for the service on the Cloudera Manager **Home** page. You *should not* explicitly restart the service after upgrading KEYTRUSTEE parcels because it automatically restarts at the appropriate point during the CDH upgrade. If you do restart the service, it will fail until after the CDH has been upgraded.

Upgrading 5.x to 6.x Parcels for Thales HSM KMS

To upgrade 5.x to 6.x parcels for Thales HSM KMS:

1. Go to **Hosts > Parcels**.
2. Click **Configuration** and add your internal repository to the **Remote Parcel Repository URLs** section. See [Configuring Cloudera Manager to Use an Internal Remote Parcel Repository](#) on page 192 for more information.
3. Click **Save Changes**.
4. Download, distribute, and activate the KEYTRUSTEE parcel for the version to which you are upgrading. See [Parcels](#) for detailed instructions on using parcels to install or upgrade components.
5. If the Thales HSM KMS already exists, then before upgrading to Cloudera Manager 6.0.0, you must change the privileged Thales HSM KMS port; the recommended port is 11501. The non-privileged port default is 9000, and does not need to be changed.

To change the privileged port, log into the Thales HSM KMS machine(s), and run the following commands:

```
# sudo /opt/nfast/bin/config-serverstartup --enable-tcp --enable-privileged-tcp
--privport=11501
[server_settings] change successful; you must restart the hardserver for this to take
effect
# sudo /opt/nfast/sbin/init.d-ncipher restart
-- Running shutdown script 90ncsnmpd

-- Running shutdown script 60raserv

...

'ncsnmpd' server now running
```

6. Restart the HSM KMS service (**HSM KMS service > Actions > Restart**).



Note: After you activate the KEYTRUSTEE parcel, Cloudera Manager displays the **Restart** button for the service on the Cloudera Manager **Home** page. You *should not* explicitly restart the service after upgrading KEYTRUSTEE parcels because it automatically restarts at the appropriate point during the CDH upgrade. If you do restart the service, it will fail until after the CDH has been upgraded.

Upgrading HSM KMS Using Packages

For customers using the Thales HSM KMS, do not use these upgrade steps. Instead, use [Upgrading 5.x to 6.x Packages for Thales HSM KMS](#) on page 173.

To upgrade an HSM KMS using packages:

1. After [Setting Up an Internal Repository](#) on page 171, configure the HSM KMS host to use the repository. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.
2. Add the CDH repository. See [Setting Up an Internal Repository](#) on page 171 for instructions. If you want to create an internal CDH repository, see [Configuring a Local Package Repository](#) on page 184.
3. Upgrade the keytrustee-keyprovider package using the appropriate command for your operating system:

- **RHEL-compatible**

```
sudo yum install keytrustee-keyprovider
```

- **SLES**

```
sudo zypper install keytrustee-keyprovider
```

- **Ubuntu or Debian**

```
sudo apt-get install keytrustee-keyprovider
```

4. Restart the HSM KMS service (**HSM KMS service** > **Actions** > **Restart**).



Note: After you activate the KEYTRUSTEE parcel, Cloudera Manager displays the **Restart** button for the service on the Cloudera Manager **Home** page. You *should not* explicitly restart the service after upgrading KEYTRUSTEE parcels because it automatically restarts at the appropriate point during the CDH upgrade. If you do restart the service, it will fail until after the CDH has been upgraded.

Upgrading 5.x to 6.x Packages for Thales HSM KMS

To upgrade 5.x to 6.x packages for Thales HSM KMS:

1. After [Setting Up an Internal Repository](#) on page 171, configure the HSM KMS host to use the repository. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.
2. Add the CDH repository. See [Setting Up an Internal Repository](#) on page 171 for instructions. If you want to create an internal CDH repository, see [Configuring a Local Package Repository](#) on page 184.
3. Upgrade the keytrustee-keyprovider package using the appropriate command for your operating system:

- **RHEL-compatible**

```
sudo yum install keytrustee-keyprovider
```

- **SLES**

```
sudo zypper install keytrustee-keyprovider
```

- **Ubuntu or Debian**

```
sudo apt-get install keytrustee-keyprovider
```

4. If the Thales HSM KMS already exists, then before upgrading to Cloudera Manager 6.0.0, you must change the privileged Thales HSM KMS port; the recommended port is 11501. The non-privileged port default is 9000, and does not need to be changed.

To change the privileged port, log into the Thales HSM KMS machine(s), and run the following commands:

```
# sudo /opt/nfast/bin/config-serverstartup --enable-tcp --enable-privileged-tcp
--privport=11501
[server_settings] change successful; you must restart the hardserver for this to take
```

Upgrading Cloudera Navigator Data Encryption

```
effect
# sudo /opt/nfast/sbin/init.d-ncipher restart
-- Running shutdown script 90ncsnmpd

-- Running shutdown script 60raserv

...

'ncsnmpd' server now running
```

5. Restart the HSM KMS service (**HSM KMS service > Actions > Restart**).



Note: After you activate the KEYTRUSTEE parcel, Cloudera Manager displays the **Restart** button for the service on the Cloudera Manager **Home** page. You *should not* explicitly restart the service after upgrading KEYTRUSTEE parcels because it automatically restarts at the appropriate point during the CDH upgrade. If you do restart the service, it will fail until after the CDH has been upgraded.

Upgrading Cloudera Navigator Encrypt

Setting Up an Internal Repository

You must create an internal repository to install or upgrade the Cloudera Navigator data encryption components. For instructions on creating internal repositories (including Cloudera Manager, CDH, and Cloudera Navigator encryption components), see the following topics:

- [Configuring a Local Parcel Repository](#) on page 189
- [Configuring a Local Package Repository](#) on page 184

Upgrading Navigator Encrypt (RHEL-Compatible)

Before you begin the upgrade process, refer to [Product Compatibility Matrix for Cloudera Navigator Encryption](#) and ensure that you have the minimum requisite operating system version(s) installed.

1. Install the Cloudera Repository

Add the internal repository you created. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.

Import the GPG key by running the following command:

```
sudo rpm --import http://repo.example.com/path/to/gpg_gazzang.asc
```

2. Stop Navigator Encrypt

Stop the Navigator Encrypt service:

```
sudo service navencrypt-mount stop
```

For RHEL 7, use `systemctl` instead:

```
sudo systemctl stop navencrypt-mount
```

3. Upgrade Navigator Encrypt

Upgrade the Navigator Encrypt client using `yum`:

```
sudo yum update navencrypt
```

4. Start Navigator Encrypt

Start the Navigator Encrypt service:

```
sudo service navencrypt-mount start
```

For RHEL 7, use `systemctl` instead:

```
sudo systemctl start navencrypt-mount
```

5. If using an RSA master key file, then you should change the master key to use OAEP padding:

```
# navencrypt key --change --rsa-oaep
...
>> Choose NEW MASTER key type:
  1) Passphrase (single)
  2) Passphrase (dual)
  3) RSA private key
Select: 3
Type MASTER RSA key file:
Type MASTER RSA key passphrase:
```

To check the type of padding currently in use:

```
# navencrypt key --get-rsa-padding
Type your Master key
Type MASTER RSA key file:
Type MASTER RSA key passphrase:

Verifying Master Key against keytrustee (wait a moment)...
RSA_PKCS1_OAEP_PADDING
```

Upgrading Navigator Encrypt (SLES)**1. Install the Cloudera Repository**

Add the internal repository you created. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.

Import the GPG key by running the following command:

```
sudo rpm --import http://repo.example.com/path/to/gpg_gazzang.asc
```

2. Stop Navigator Encrypt

Stop the Navigator Encrypt service:

```
sudo service navencrypt-mount stop
```

3. Upgrade the Kernel Module Package (KMP)

```
sudo zypper update cloudera-navencryptfs-kmp-<kernel_flavor>
```

Replace `kernel_flavor` with the kernel flavor for your system. Navigator Encrypt supports the default, xen, and ec2 kernel flavors.

4. Upgrade the Navigator Encrypt Client

Upgrade Navigator Encrypt:

```
sudo zypper update navencrypt
```

5. Enable Unsupported Modules

Upgrading Cloudera Navigator Data Encryption

Edit `/etc/modprobe.d/unsupported-modules` and set `allow_unsupported_modules` to 1. For example:

```
#
# Every kernel module has a flag 'supported'. If this flag is not set loading
# this module will taint your kernel. You will not get much help with a kernel
# problem if your kernel is marked as tainted. In this case you firstly have
# to avoid loading of unsupported modules.
#
# Setting allow_unsupported_modules 1 enables loading of unsupported modules
# by modprobe, setting allow_unsupported_modules 0 disables it. This can
# be overridden using the --allow-unsupported-modules command line switch.
allow_unsupported_modules 1
```

6. Start Navigator Encrypt

Start the Navigator Encrypt service:

```
sudo service navencrypt-mount start
```

7. If using an RSA master key file, then you should change the master key to use OAEP padding:

```
# navencrypt key --change --rsa-oaep
...
>> Choose NEW MASTER key type:
1) Passphrase (single)
2) Passphrase (dual)
3) RSA private key
Select: 3
Type MASTER RSA key file:
Type MASTER RSA key passphrase:
```

To check the type of padding currently in use:

```
# navencrypt key --get-rsa-padding
Type your Master key
Type MASTER RSA key file:
Type MASTER RSA key passphrase:

Verifying Master Key against keytrustee (wait a moment)...
RSA_PKCS1_OAEP_PADDING
```

Upgrading Navigator Encrypt (Debian or Ubuntu)

1. Install the Cloudera Repository

Add the internal repository you created. See [Configuring Hosts to Use the Internal Repository](#) on page 188 for more information.

• Ubuntu

```
echo "deb http://repo.example.com/path/to/ubuntu/stable $DISTRIB_CODENAME main" | sudo
tee -a /etc/apt/sources.list
```

• Debian

```
echo "deb http://repo.example.com/path/to/debian/stable $DISTRIB_CODENAME main" | sudo
tee -a /etc/apt/sources.list
```

Import the GPG key by running the following command:

```
wget -O - http://repo.example.com/path/to/gpg_gazzang.asc | apt-key add -
```

Update the repository index with `apt-get update`.

2. Stop Navigator Encrypt

Stop the Navigator Encrypt service:

```
sudo service navencrypt-mount stop
```

3. Upgrade the Navigator Encrypt Client

Upgrade Navigator Encrypt:

```
sudo apt-get install navencrypt
```

4. Start Navigator Encrypt

Start the Navigator Encrypt service:

```
sudo service navencrypt-mount start
```

5. If using an RSA master key file, then you should change the master key to use OAEP padding:

```
# navencrypt key --change --rsa-oaep
...
>> Choose NEW MASTER key type:
  1) Passphrase (single)
  2) Passphrase (dual)
  3) RSA private key
Select: 3
Type MASTER RSA key file:
Type MASTER RSA key passphrase:
```

To check the type of padding currently in use:

```
# navencrypt key --get-rsa-padding
Type your Master key
Type MASTER RSA key file:
Type MASTER RSA key passphrase:

Verifying Master Key against keytrustee (wait a moment)...
RSA_PKCS1_OAEP_PADDING
```

Best Practices for Upgrading Navigator Encrypt Hosts

Following are some best practices for upgrading operating systems (OS) and kernels on hosts that have Navigator Encrypt installed:

- Make sure that the version you are upgrading to is supported by Navigator Encrypt. See the product compatibility matrix for [Product Compatibility Matrix for Cloudera Navigator Encryption](#) for more information.
- Always test upgrades in a development or testing environment before upgrading production hosts.
- If possible, upgrade the entire operating system instead of only upgrading the kernel.
- If you need to upgrade the kernel only, make sure that your OS version supports the kernel version to which you are upgrading.
- Always back up the `/etc/navencrypt` directory before upgrading. If you have problems accessing encrypted data after upgrading the OS or kernel, restore `/etc/navencrypt` from your backup and try again.

Migrating from the Cloudera Manager Embedded PostgreSQL Database Server to an External PostgreSQL Database

Cloudera Manager provides an embedded PostgreSQL database server for demonstration and proof of concept deployments when creating a cluster. To remind users that this embedded database is not suitable for production, Cloudera Manager displays the banner text: "You are running Cloudera Manager in non-production mode, which uses an embedded PostgreSQL database. Switch to using a supported external database before moving into production."

If, however, you have already used the embedded database, and you are unable to redeploy a fresh cluster, then you *must* migrate to an external PostgreSQL database.



Note: This procedure does *not* describe how to migrate to a database server other than PostgreSQL. Moving databases from one database server to a different type of database server is a complex process that requires modification of the schema and matching the data in the database tables to the new schema. It is strongly recommended that you engage with Cloudera Professional Services if you wish to perform a migration to an external database server other than PostgreSQL.

Prerequisites

Before migrating the Cloudera Manager embedded PostgreSQL database to an external PostgreSQL database, ensure that your setup meets the following conditions:

- The external PostgreSQL database server is running.
- The database server is configured to accept remote connections.
- The database server is configured to accept user logins using md5.
- No one has manually created any databases in the external database server for roles that will be migrated.



Note: To view a list of databases in the external database server (requires default superuser permission):

```
sudo -u postgres psql -l
```

- All health issues with your cluster have been resolved.

For details about configuring the database server, see [Configuring and Starting the PostgreSQL Server](#).



Important: Only perform the steps in [Configuring and Starting the PostgreSQL Server](#). Do *not* proceed with the creation of databases as described in the subsequent section.

For large clusters, Cloudera recommends running your database server on a dedicated host. Engage Cloudera Professional Services or a certified database administrator to correctly tune your external database server.

Identify Roles that Use the Embedded Database Server

Before you can migrate to another database server, you must first identify the databases using the embedded database server. When the Cloudera Manager Embedded Database server is initialized, it creates the Cloudera Manager database and databases for roles in the Management Services. The Installation Wizard (which runs automatically the first time you log in to Cloudera Manager) or **Add Service** action for a cluster creates additional databases for roles when run. It is in this context that you identify which roles are used in the embedded database server.

To identify which roles are using the Cloudera Manager embedded database server:

1. Obtain and save the `cloudera-scm` superuser password from the embedded database server. You will need this password in subsequent steps:

```
head -1 /var/lib/cloudera-scm-server-db/data/generated_password.txt
```

2. Make a list of all services that are using the embedded database server. Then, after determining which services are not using the embedded database server, remove those services from the list. The `scm` database must remain in your list. Use the following table as a guide:

Table 4: Cloudera Manager Embedded Database Server Databases

Service	Role	Default Database Name	Default Username
Cloudera Manager Server		scm	scm
Cloudera Management Service	Activity Monitor	amon	amon
Hive	Hive Metastore Server	hive	hive
Hue	Hue Server	hue	7uu7uu7uhue
Cloudera Management Service	Navigator Audit Server	nav	nav
Cloudera Management Service	Navigator Metadata Server	navms	navms
Oozie	Oozie Server	oozie_oozie_server	oozie_oozie_server
Cloudera Management Service	Reports Manager	rman	rman
Sentry	Sentry Server	sentry	sentry

3. Verify which roles are using the embedded database. Roles using the embedded database server always use port 7432 (the default port for the embedded database) on the Cloudera Manager Server host.

For Cloudera Management Services:

- a. Select **Cloudera Management Service > Configuration**, and type "7432" in the **Search** field.
- b. Confirm that the hostname for the services being used is the same hostname used by the Cloudera Manager Server.



Note:

If any of the following fields contain the value "7432", then the service is using the embedded database:

- **Activity Monitor**
- **Navigator Audit Server**
- **Navigator Metadata Server**
- **Reports Manager**

For the Oozie Service:

1. Select **Oozie service > Configuration**, and type "7432" in the **Search** field.
2. Confirm that the hostname is the Cloudera Manager Server.

For Hive, Hue, and Sentry Services:

1. Select the specific service > **Configuration**, and type "database host" in the **Search** field.
 2. Confirm that the hostname is the Cloudera Manager Server.
 3. In the **Search** field, type "database port" and confirm that the port is 7432.
 4. Repeat these steps for each of the services (Hive, Hue and Sentry).
4. Verify the database names in the embedded database server match the database names on your list (Step 2). Databases that exist on the database server and not used by their roles do *not* need to be migrated. This step is to confirm that your list is correct.



Note: Do not add the `postgres`, `template0`, or `template1` databases to your list. These are used only by the PostgreSQL server.

```
psql -h localhost -p 7432 -U cloudera-scm -l
```

Password for user cloudera-scm: <password>

Name Access	Owner	List of databases			Ctype
		Encoding	Collate		
amon	amon	UTF8	en_US.UTF8	en_US.UTF8	
hive	hive	UTF8	en_US.UTF8	en_US.UTF8	
hue	hue	UTF8	en_US.UTF8	en_US.UTF8	
nav	nav	UTF8	en_US.UTF8	en_US.UTF8	
navms	navms	UTF8	en_US.UTF8	en_US.UTF8	
oozie_oozie_server	oozie_oozie_server	UTF8	en_US.UTF8	en_US.UTF8	
postgres	cloudera-scm	UTF8	en_US.UTF8	en_US.UTF8	
rman	rman	UTF8	en_US.UTF8	en_US.UTF8	
scm	scm	UTF8	en_US.UTF8	en_US.UTF8	
sentry	sentry	UTF8	en_US.UTF8	en_US.UTF8	
template0	cloudera-scm	UTF8	en_US.UTF8	en_US.UTF8	
=c/"cloudera-scm"					
template1	cloudera-scm	UTF8	en_US.UTF8	en_US.UTF8	
=c/"cloudera-scm"					
(12 rows)					

You should now have a list of all roles and database names that use the embedded database server, and are ready to proceed with the migration of databases from the embedded database server to the external PostgreSQL database server.

Migrate Databases from the Embedded Database Server to the External PostgreSQL Database Server

While performing this procedure, ensure that the Cloudera Manager Agents remain running on all hosts. Unless otherwise specified, when prompted for a password use the `cloudera-scm` password.



Note: After completing this migration, you cannot delete the `cloudera-scm postgres` superuser unless you remove the access privileges for the migrated databases. Minimally, you should change the `cloudera-scm postgres` superuser password.

1. In Cloudera Manager, stop the cluster services identified as using the embedded database server (see [Identify Roles that Use the Embedded Database Server](#) on page 178). Refer to [Starting, Stopping, and Restarting Services](#) for details about how to stop cluster services. Be sure to stop the Cloudera Management Service as well. Also be sure to stop any services with dependencies on these services. The remaining CDH services will continue to run without downtime.



Note: If you do not stop the services from within Cloudera Manager before stopping Cloudera Manager Server from the command line, they will continue to run and maintain a network connection to the embedded database server. If this occurs, then the embedded database server will ignore any command line stop commands (Step 2) and require that you manually kill the process, which in turn causes the services to crash instead of stopping cleanly.

2. Navigate to **Hosts > All Hosts**, and make note of the number of roles assigned to hosts. Also take note whether or not they are in a commissioned state. You will need this information later to validate that your `scm` database was migrated correctly.
3. Stop the Cloudera Manager Server. To stop the server:

```
sudo service cloudera-scm-server stop
```

4. Obtain and save the embedded database superuser password (you will need this password in subsequent steps) from the `generated_password.txt` file:

```
head -1 /var/lib/cloudera-scm-server-db/data/generated_password.txt
```

5. Export the PostgreSQL user roles from the embedded database server to ensure the correct users, permissions, and passwords are preserved for database access. Passwords are exported as an `md5sum` and are not visible in plain text. To export the database user roles (you will need the `cloudera-scm` user password):

```
pg_dumpall -h localhost -p 7432 -U cloudera-scm -v --roles-only -f  
"/var/tmp/cloudera_user_roles.sql"
```

6. Edit `/var/tmp/cloudera_user_roles.sql` to remove any `CREATE ROLE` and `ALTER ROLE` commands for databases not in your list. Leave the entries for `cloudera-scm` untouched, because this user role is used during the database import.
7. Export the data from each of the databases on your list you created in [Identify Roles that Use the Embedded Database Server](#) on page 178:

```
pg_dump -F c -h localhost -p 7432 -U cloudera-scm [database_name] >  
/var/tmp/[database_name]_db_backup-$(date +%m-%d-%Y).dump
```

Following is a sample data export command for the `scm` database:

```
pg_dump -F c -h localhost -p 7432 -U cloudera-scm scm > /var/tmp/scm_db_backup-$(date  
+%m-%d-%Y).dump
```

Password:

8. Stop and disable the embedded database server:

```
service cloudera-scm-server-db stop  
chkconfig cloudera-scm-server-db off
```

Confirm that the embedded database server is stopped:

```
netstat -at | grep 7432
```

9. Back up the Cloudera Manager Server database configuration file:

```
cp /etc/cloudera-scm-server/db.properties /etc/cloudera-scm-server/db.properties.embedded
```

Migrating from the Cloudera Manager Embedded PostgreSQL Database Server to an External PostgreSQL Database

- 10 Copy the file `/var/tmp/cloudera_user_roles.sql` and the database dump files from the embedded database server host to `/var/tmp` on the external database server host:

```
cd /var/tmp
scp cloudera_user_roles.sql *.dump <user>@<postgres-server>:/var/tmp
```

- 11 Import the PostgreSQL user roles into the external database server.

The external PostgreSQL database server superuser password is required to import the user roles. If the superuser role has been changed, you will be prompted for the username and password.



Note: Only run the command that applies to your context; do *not* execute both commands.

- To import users when using the default PostgreSQL superuser role:

```
sudo -u postgres psql -f /var/tmp/cloudera_user_roles.sql
```

- To import users when the superuser role has been changed:

```
psql -h <database-hostname> -p <database-port> -U <superuser> -f
/var/tmp/cloudera_user_roles.sql
```

For example:

```
psql -h pg-server.example.com -p 5432 -U postgres -f /var/tmp/cloudera_user_roles.sql
```

```
Password for user postgres
```

- 12 Import the Cloudera Manager database on the external server. First copy the database dump files from the Cloudera Manager Server host to your external PostgreSQL database server, and then import the database data:



Note: To successfully run the `pg_restore` command, there must be an existing database on the database server to complete the connection; the existing database will *not* be modified. If the `-d <existing-database>` option is not included, then the `pg_restore` command will fail.

```
pg_restore -C -h <database-hostname> -p <database-port> -d <existing-database> -U
cloudera-scm -v <data-file>
```

Repeat this import for each database.

The following example is for the `scm` database:

```
pg_restore -C -h pg-server.example.com -p 5432 -d postgres -U cloudera-scm -v
/var/tmp/scm_server_db_backup-20180312.dump
```

```
pg_restore: connecting to database for restore
Password:
```

- 13 Update the Cloudera Manager Server database configuration file to use the external database server. Edit the `/etc/cloudera-scm-server/db.properties` file as follows:

- a. Update the `com.cloudera.cmf.db.host` value with the hostname and port number of the external database server.
- b. Change the `com.cloudera.cmf.db.setupType` value from "EMBEDDED" to "EXTERNAL".

14 Start the Cloudera Manager Server and confirm it is working:

```
service cloudera-scm-server start
```

Note that if you start the Cloudera Manager GUI at this point, it may take up to five minutes after executing the start command before it becomes available.

In Cloudera Manager Server, navigate to **Hosts > All Hosts** and confirm the number of roles assigned to hosts (this number should match what you found in Step 2); also confirm that they are in a commissioned state that matches what you observed in Step 2.

15 Update the role configurations to use the external database hostname and port number. Only perform this task for services where the database has been migrated.

For Cloudera Management Services:

1. Select **Cloudera Management Service > Configuration**, and type "7432" in the **Search** field.
2. Change any database hostname properties from the embedded database to the external database hostname and port number.
3. Click **Save Changes**.

For the Oozie Service:

- a. Select **Oozie service > Configuration**, and type "7432" in the **Search** field.
- b. Change any database hostname properties from the embedded database to the external database hostname and port number.
- c. Click **Save Changes**.

For Hive, Hue, and Sentry Services:

1. Select the specific service > **Configuration**, and type "database host" in the **Search** field.
2. Change the hostname from the embedded database name to the external database hostname.
3. Click **Save Changes**.

16 Start the **Cloudera Management Service** and confirm that all management services are up and no health tests are failing.

17 Start all Services via the Cloudera Manager web UI. This should start all services that were stopped for the database migration. Confirm that all services are up and no health tests are failing.

18 On the embedded database server host, remove the embedded PostgreSQL database server:

- a. Make a backup of the `/var/lib/cloudera-scm-server-db/data` directory:

```
tar czvf /var/tmp/embedded_db_data_backup-$(date +%m-%d-%Y).tgz
/var/lib/cloudera-scm-server-db/data
```

- b. Remove the embedded database package:

For RHEL/SLES:

```
rpm --erase cloudera-manager-server-db-2
```

For Debian/Ubuntu:

```
apt-get remove cloudera-manager-server-db-2
```

- c. Delete the `/var/lib/cloudera-scm-server-db/data` directory.

Configuring a Local Package Repository

You can create a package repository for Cloudera Manager either by hosting an internal web repository or by manually copying the repository files to the Cloudera Manager Server host for distribution to Cloudera Manager Agent hosts.

Creating a Permanent Internal Repository

The following sections describe how to create a permanent internal repository using Apache HTTP Server:

Setting Up a Web server

To host an internal repository, you must install or use an existing Web server on an internal host that is reachable by the Cloudera Manager host, and then download the repository files to the Web server host. The examples in this section use Apache HTTP Server as the Web server. If you already have a Web server in your organization, you can skip to [Downloading and Publishing the Package Repository](#) on page 184.

1. Install Apache HTTP Server:

RHEL / CentOS

```
sudo yum install httpd
```

SLES

```
sudo zypper install httpd
```

Debian

```
sudo apt-get install httpd
```

2. Start Apache HTTP Server:

RHEL 7

```
sudo systemctl start httpd
```

RHEL 6 or lower

```
sudo service httpd start
```

SLES 12, Ubuntu 16 or later, Debian 8

```
sudo systemctl start apache2
```

SLES 11, Ubuntu 14.04, Debian 7 or lower

```
sudo service apache2 start
```

Downloading and Publishing the Package Repository

1. Download the package repository for the product you want to install:

Cloudera Manager 6

To download the files for the latest Cloudera Manager 6.2 release, run the following commands on the Web server host. Replace `<operating_system>` with the operating system you are using (redhat7, redhat6, sles12, ubuntu1604, or ubuntu1804):

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/cm6/6.2.0/<operating_system>/ -P /var/www/html/cloudera-repos
sudo wget https://archive.cloudera.com/cm6/6.2.0/allkeys.asc -P
/var/www/html/cloudera-repos/cm6/6.2.0/
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cm6
```

If you want to create a repository for a different Cloudera Manager 6 release, replace 6.2.0 with the CDH 6 version that you want. For more information, see [Cloudera Manager 6 Version and Download Information](#).

CDH 6

To download the files for the latest CDH 6.2 release, run the following commands on the Web server host. Replace `<operating_system>` with the operating system you are using (redhat7, redhat6, sles12, ubuntu1604, or ubuntu1804):

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/cdh6/6.2.0/<operating_system>/ -P
/var/www/html/cloudera-repos
```

```
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/gplextras6/6.2.0/<operating_system>/ -P
/var/www/html/cloudera-repos
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cdh6
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/gplextras6
```

If you want to create a repository for a different CDH 6 release, replace 6.2.0 with the CDH 6 version that you want. For more information, see [CDH 6 Download Information](#).

Cloudera Manager 5

To download the files for a Cloudera Manager release, download the repository tarball for your operating system.

```
sudo mkdir -p /var/www/html/cloudera-repos/cm5
```

Redhat/Centos:

```
wget https://archive.cloudera.com/cm5/repo-as-tarball/5.14.4/cm5.14.4-centos7.tar.gzd
```

```
tar xvfz cm5.14.4-centos7.tar.gz -C /var/www/html/cloudera-repos/cm5 --strip-components=1
```

Debian:

```
wget https://archive.cloudera.com/cm5/repo-as-tarball/5.14.4/cm5.14.4-debian-jessie.tar.gz
```

```
tar xvfz cm5.14.4-debian-jessie.tar.gz -C /var/www/html/cloudera-repos/cm5
--strip-components=1
```

SLES:

```
wget https://archive.cloudera.com/cm5/repo-as-tarball/5.14.4/cm5.14.4-sles12.tar.gz
```

```
tar xvfz cm5.14.4-sles12.tar.gz -C /var/www/html/cloudera-repos/cm5 --strip-components=1
```

Ubuntu:

```
wget https://archive.cloudera.com/cm5/repo-as-tarball/5.14.4/cm5.14.4-ubuntu16-04.tar.gz
```

```
tar xvfz cm5.14.4-ubuntu16-04.tar.gz -C /var/www/html/cloudera-repos/cm5  
--strip-components=1
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cm5
```

If you want to create a repository for a different Cloudera Manager release or operating system, start in the [repo-as-tarball](#) parent directory, select the Cloudera Manager version you want to use, and then copy the `.tar.gz` link for your operating system.

CDH 5

To download the files for a CDH release, download the repository tarball for your operating system.

```
sudo mkdir -p /var/www/html/cloudera-repos/cdh5
```

Redhat/Centos:

```
wget https://archive.cloudera.com/cdh5/repo-as-tarball/5.14.4/cdh5.14.4-centos7.tar.gz
```

```
tar xvfz cdh5.14.4-centos7.tar.gz -C /var/www/html/cloudera-repos/cdh5  
--strip-components=1
```

Debian:

```
wget  
https://archive.cloudera.com/cdh5/repo-as-tarball/5.14.4/cdh5.14.4-debian-jessie.tar.gz
```

```
tar xvfz cdh5.14.4-debian-jessie.tar.gz -C /var/www/html/cloudera-repos/cdh5  
--strip-components=1
```

SLES:

```
wget https://archive.cloudera.com/cdh5/repo-as-tarball/5.14.4/cdh5.14.4-sles12.tar.gz
```

```
tar xvfz cdh5.14.4-sles12.tar.gz -C /var/www/html/cloudera-repos/cdh5 --strip-components=1
```

Ubuntu:

```
wget https://archive.cloudera.com/cdh5/repo-as-tarball/5.14.4/cdh5.14.4-ubuntu16-04.tar.gz
```

```
tar xvfz cdh5.14.4-ubuntu16-04.tar.gz -C /var/www/html/cloudera-repos/cdh5  
--strip-components=1
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cdh5
```

If you want to create a repository for a different CDH release or operating system, start in the [repo-as-tarball](#) parent directory, select the CDH version you want to use, and then copy the `.tar.gz` link for your operating system.

Apache Accumulo for CDH

To download the files for an Accumulo release for CDH, run the following commands on the Web server host. Replace `<operating_system>` with the OS you are using (redhat, sles, debian, or ubuntu):

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/accumulo-c5/<operating_system>/ -P
/var/www/html/cloudera-repos
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/accumulo-c5
```

Cloudera Navigator Key Trustee Server

Go to the Key Trustee Server [download page](#). Select **Packages** from the **CHOOSE DOWNLOAD TYPE** drop-down menu, select your operating system from the **CHOOSE AN OS** drop-down menu, and then click **DOWNLOAD NOW**. This downloads the Key Trustee Server package files in a `.tar.gz` file. Copy the file to your Web server, and extract the files with the `tar xvfz filename.tar.gz` command. This example uses Key Trustee Server 5.14.0:

```
sudo mkdir -p /var/www/html/cloudera-repos/keytrustee-server
sudo tar xvfz /path/to/keytrustee-server-5.14.0-parcels.tar.gz -C
/var/www/html/cloudera-repos/keytrustee-server --strip-components=1
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/keytrustee-server
```

Cloudera Navigator Key Trustee KMS and HSM KMS

Note: Cloudera Navigator HSM KMS is included in the Key Trustee KMS packages.

Go to the Key Trustee KMS [download page](#). Select **Package** from the **CHOOSE DOWNLOAD TYPE** drop-down menu, select your operating system from the **OPERATING SYSTEM** drop-down menu, and then click **DOWNLOAD NOW**. This downloads the Key Trustee KMS package files in a `.tar.gz` file. Copy the file to your Web server, and extract the files with the `tar xvfz filename.tar.gz` command. This example uses Key Trustee KMS 5.14.0:

```
sudo mkdir -p /var/www/html/cloudera-repos/keytrustee-kms
sudo tar xvfz /path/to/keytrustee-kms-5.14.0-parcels.tar.gz -C
/var/www/html/cloudera-repos/keytrustee-kms --strip-components=1
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/keytrustee-kms
```

2. Visit the Repository URL `http://<web_server>/cloudera-repos/` in your browser and verify the files you downloaded are present. If you do not see anything, your Web server may have been configured to not show indexes.

Creating a Temporary Internal Repository

You can quickly create a temporary remote repository to deploy packages on a one-time basis. Cloudera recommends using the same host that runs Cloudera Manager, or a gateway host. This example uses [Python SimpleHTTPServer](#) as the Web server to host the `/var/www/html` directory, but you can use a different directory.

1. Download the repository you need following the instructions in [Downloading and Publishing the Package Repository](#) on page 184.
2. Determine a port that your system is not listening on. This example uses port 8900.

Configuring a Local Package Repository

3. Start a Python SimpleHTTPServer in the /var/www/html directory:

```
cd /var/www/html
python -m SimpleHTTPServer 8900
```

Serving HTTP on 0.0.0.0 port 8900 ...

4. Visit the Repository URL `http://<web_server>:8900/cloudera-repos/` in your browser and verify the files you downloaded are present.

Configuring Hosts to Use the Internal Repository

After establishing the repository, modify the client configuration to use it:

OS	Procedure
RHEL compatible	<p>Create <code>/etc/yum.repos.d/cloudera-repo.repo</code> files on cluster hosts with the following content, where <code><web_server></code> is the hostname of the Web server:</p> <pre>[cloudera-repo] name=cloudera-repo baseurl=http://<web_server>/cm/5 enabled=1 gpgcheck=0</pre>
SLES	<p>Use the <code>zypper</code> utility to update client system repository information by issuing the following command:</p> <pre>zypper addrepo http://<web_server>/cm <alias></pre>
Ubuntu	<p>Create <code>/etc/apt/sources.list.d/cloudera-repo.list</code> files on all cluster hosts with the following content, where <code><web_server></code> is the hostname of the Web server:</p> <pre>deb http://<web_server>/cm <codename> <components></pre> <p>You can find the <code><codename></code> and <code><components></code> variables in the <code>./conf/distributions</code> file in the repository.</p> <p>After creating the <code>.list</code> file, run the following command:</p> <pre>sudo apt-get update</pre>

Configuring a Local Parcel Repository

You can create a parcel repository for Cloudera Manager either by hosting an internal Web repository or by manually copying the repository files to the Cloudera Manager Server host for distribution to Cloudera Manager Agent hosts.

Using an Internally Hosted Remote Parcel Repository

The following sections describe how to use an internal Web server to host a parcel repository:

Setting Up a Web Server

To host an internal repository, you must install or use an existing Web server on an internal host that is reachable by the Cloudera Manager host, and then download the repository files to the Web server host. The examples on this page use Apache HTTP Server as the Web server. If you already have a Web server in your organization, you can skip to [Downloading and Publishing the Parcel Repository](#) on page 190.

1. Install Apache HTTP Server:

RHEL / CentOS

```
sudo yum install httpd
```

SLES

```
sudo zypper install httpd
```

Debian

```
sudo apt-get install httpd
```

- 2.



Warning: Skipping this step could result in an error message **Hash verification failed** when trying to download the parcel from a local repository, especially in Cloudera Manager 6 and higher.

Edit the Apache HTTP Server configuration file (/etc/httpd/conf/httpd.conf by default) to add or edit the following line in the <IfModule mime_module> section:

```
AddType application/x-gzip .gz .tgz .parcel
```

If the <IfModule mime_module> section does not exist, you can add it in its entirety as follows:



Note: This example configuration was modified from the default configuration provided after installing Apache HTTP Server on RHEL 7.

```
<IfModule mime_module>
#
# TypesConfig points to the file containing the list of mappings from
# filename extension to MIME-type.
#
TypesConfig /etc/mime.types

#
# AddType allows you to add to or override the MIME configuration
# file specified in TypesConfig for specific file types.
#
AddType application/x-gzip .gz .tgz
#
```

Configuring a Local Parcel Repository

```
# AddEncoding allows you to have certain browsers uncompress
# information on the fly. Note: Not all browsers support this.
#
#AddEncoding x-compress .Z
#AddEncoding x-gzip .gz .tgz
#
# If the AddEncoding directives above are commented-out, then you
# probably should define those extensions to indicate media types:
#
AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz .parcel

#
# AddHandler allows you to map certain file extensions to "handlers":
# actions unrelated to filetype. These can be either built into the server
# or added with the Action directive (see below)
#
# To use CGI scripts outside of ScriptAliased directories:
# (You will also need to add "ExecCGI" to the "Options" directive.)
#
#AddHandler cgi-script .cgi

# For type maps (negotiated resources):
#AddHandler type-map var

#
# Filters allow you to process content before it is sent to the client.
#
# To parse .shtml files for server-side includes (SSI):
# (You will also need to add "Includes" to the "Options" directive.)
#
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
</IfModule>
```

3. Start Apache HTTP Server:

RHEL 7

```
sudo systemctl start httpd
```

RHEL 6 or lower

```
sudo service httpd start
```

SLES 12, Ubuntu 16 or later, Debian 8

```
sudo systemctl start apache2
```

SLES 11, Ubuntu 14.04, Debian 7 or lower

```
sudo service apache2 start
```

Downloading and Publishing the Parcel Repository

1. Download manifest.json and the parcel files for the product you want to install:

CDH 6

Apache Impala, Apache Kudu, Apache Spark 2, and Cloudera Search are included in the CDH parcel. To download the files for the latest CDH 6.2 release, run the following commands on the Web server host:

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/cdh6/6.2.0/parcels/ -P /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/gplextras6/6.2.0/parcels/ -P /var/www/html/cloudera-repos
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cdh6
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/gplextras6
```

If you want to create a repository for a different CDH 6 release, replace 6.2.0 with the CDH 6 version that you want. For more information, see [CDH 6 Download Information](#).

CDH 5

Impala, Kudu, Spark 1, and Search are included in the CDH parcel. To download the files for a CDH release (CDH 5.14.4 in this example), run the following commands on the Web server host:

```
sudo mkdir -p /var/www/html/cloudera-repos
```

```
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/cdh5/parcels/5.14.4/ -P /var/www/html/cloudera-repos
```

```
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/gplextras5/parcels/5.14.4/ -P /var/www/html/cloudera-repos
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cdh5
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/gplextras5
```

If you want to create a repository for a different CDH release, replace 5.14.4 with the CDH version that you want. For more information, see [CDH Download Information](#).

Apache Accumulo for CDH

To download the files for an Accumulo release for CDH (Accumulo 1.7.2 in this example), run the following commands on the Web server host:

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/accumulo-c5/parcels/1.7.2/ -P /var/www/html/cloudera-repos
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/accumulo-c5
```

If you want to create a repository for Accumulo 1.6.0 instead, replace 1.7.2 with 1.6.0.

CDS Powered By Apache Spark 2 for CDH

To download the files for a CDS release for CDH (CDS 2.3.0.cloudera3 in this example), run the following commands on the Web server host:

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
https://archive.cloudera.com/spark2/parcels/2.3.0.cloudera3/ -P
/var/www/html/cloudera-repos
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/spark2
```

If you want to create a repository for a different CDS release, replace 2.3.0.cloudera3 with the CDS version that you want. For more information, see [CDS Powered By Apache Spark Version Information](#).

Cloudera Navigator Key Trustee Server

Go to the Key Trustee Server [download page](#). Select **Parcels** from the **CHOOSE DOWNLOAD TYPE** drop-down menu, and click **DOWNLOAD NOW**. This downloads the Key Trustee Server parcels and manifest.json files in

Configuring a Local Parcel Repository

a .tar.gz file. Copy the file to your Web server, and extract the files with the `tar xvfz filename.tar.gz` command. This example uses Key Trustee Server 5.14.0:

```
sudo mkdir -p /var/www/html/cloudera-repos/keytrustee-server
sudo tar xvfz /path/to/keytrustee-server-5.14.0-parcels.tar.gz -C
/var/www/html/cloudera-repos/keytrustee-server --strip-components=1
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/keytrustee-server
```

Cloudera Navigator Key Trustee KMS and HSM KMS



Note: Cloudera Navigator HSM KMS is included in the Key Trustee KMS parcel.

Go to the Key Trustee KMS [download page](#). Select **Parcels** from the **CHOOSE DOWNLOAD TYPE** drop-down menu, and click **DOWNLOAD NOW**. This downloads the Key Trustee KMS parcels and manifest.json files in a .tar.gz file. Copy the file to your Web server, and extract the files with the `tar xvfz filename.tar.gz` command. This example uses Key Trustee KMS 5.14.0:

```
sudo mkdir -p /var/www/html/cloudera-repos/keytrustee-kms
sudo tar xvfz /path/to/keytrustee-kms-5.14.0-parcels.tar.gz -C
/var/www/html/cloudera-repos/keytrustee-kms --strip-components=1
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/keytrustee-kms
```

Sqoop Connectors

To download the parcels for a Sqoop Connector release, run the following commands on the Web server host. This example uses the latest available Sqoop Connectors:

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories
http://archive.cloudera.com/sqoop-connectors/parcels/latest/ -P
/var/www/html/cloudera-repos
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/sqoop-connectors
```

If you want to create a repository for a different Sqoop Connector release, replace `latest` with the Sqoop Connector version that you want. You can see a list of versions in the [parcels](#) parent directory.

2. Visit the Repository URL `http://<Web_server>/cloudera-repos/` in your browser and verify the files you downloaded are present. If you do not see anything, your Web server may have been configured to not show indexes.

Configuring Cloudera Manager to Use an Internal Remote Parcel Repository

1. Use one of the following methods to open the parcel settings page:
 - **Navigation bar**
 1. Click the parcel icon in the top navigation bar or click **Hosts** and click the **Parcels** tab.
 2. Click the **Configuration** button.
 - **Menu**
 1. Select **Administration > Settings**.
 2. Select **Category > Parcels**.
2. In the **Remote Parcel Repository URLs** list, click the addition symbol to open an additional row.
3. Enter the path to the parcel. For example: `http://<web_server>/cloudera-parcels/cdh6/6.2.0/`
4. Enter a **Reason for change**, and then click **Save Changes** to commit the changes.

Using a Local Parcel Repository

To use a local parcel repository, complete the following steps:

1. Open the Cloudera Manager Admin Console and navigate to the **Parcels** page.
2. Select **Configuration** and verify that you have a **Local Parcel Repository** path set. By default, the directory is `/opt/cloudera/parcel-repo`.
3. Remove any **Remote Parcel Repository URLs** you are not using, including ones that point to Cloudera archives.
4. Add the parcel you want to use to the local parcel repository directory that you specified. For instructions on downloading parcels, see [Downloading and Publishing the Parcel Repository](#) on page 190 above.
5. In the command line, navigate to the local parcel repository directory.
6. Create a SHA1 hash for the parcel you added and save it to a file named `parcel_name.parcel.sha`.

For example, the following command generates a SHA1 hash for the parcel

`CDH-6.1.0-1.cdh6.1.0.p0.770702-el7.parcel`:

```
shasum CDH-6.1.0-1.cdh6.1.0.p0.770702-el7.parcel | awk '{ print $1 }' >
CDH-6.1.0-1.cdh6.1.0.p0.770702-el7.parcel.sha
```

7. Change the ownership of the parcel and hash files to `cloudera-scm`:

```
sudo chown -R cloudera-scm:cloudera-scm /opt/cloudera/parcel-repo/*
```

8. In the Cloudera Manager Admin Console, navigate to the **Parcels** page.
9. Click **Check for New Parcels** and verify that the new parcel appears.
10. Download, distribute, and activate the parcel.

Understanding Keystores and Truststores

Configuring Cloudera Manager Server and cluster components to use TLS/SSL requires obtaining keys, certificates, and related security artifacts. The following provides a brief overview.

Java Keystore and Truststore

All clients in a Cloudera Manager cluster configured for TLS/SSL need access to the truststore to validate certificates presented during TLS/SSL session negotiation. The certificates assure the client or server process that the issuing authority for the certificate is part of a legitimate chain of trust.

The standard Oracle Java JDK distribution includes a default **truststore** (`cacerts`) that contains root certificates for many well-known CAs, including Symantec. Rather than using the default truststore, Cloudera recommends using the **alternative truststore**, `jssecacerts`. The alternative truststore is created by copying `cacerts` to that filename (`jssecacerts`). Certificates can be added to this truststore when needed for additional roles or services. This alternative truststore is loaded by Hadoop daemons at startup.



Important: For use with Cloudera clusters, the alternative trust store—`jssecacerts`—must start as a **copy of `cacerts`** because `cacerts` contains all available default certificates needed to establish the chain of trust during the TLS/SSL handshake. After `jssecacerts` has been created, new public and private root CAs are added to it for use by the cluster. See [Generate TLS Certificates](#) for details.

The private keys are maintained in the **keystore**.



Note: For detailed information about the Java keystore and truststore, see Oracle documentation:

- [Keytool—Key and Certificate Management Tool](#)
- [JSSE Reference Guide for Java](#)

Although the keystore and truststore in some environments may comprise the same file, as configured for Cloudera Manager Server and CDH clusters, the keystore and truststore are distinct files. For Cloudera Manager Server clusters, each host should have its own keystore, while several hosts can share the same truststore. This table summarizes the general differences between keystore and the truststore in Cloudera Manager Server clusters.

Keystore	Truststore
Used by the server side of a TLS/SSL client-server connection.	Used by the client side of a TLS/SSL client-server connection.
Typically contains 1 private key for the host system.	Contains no keys of any kind.
Contains the certificate for the host's private key.	Contains root certificates for well-known public certificate authorities. May contain certificates for intermediary certificate authorities.
Password protected. Use the same password for the key and its keystore.	Password-protection not needed. However, if password has been used for the truststore, never use the same password as used for a key and keystore.
Password stored in a plaintext file read permissions granted to a specific group only (OS filesystem permissions set to 0440, <code>hadoop:hadoop</code>).	Password (if there is one for the truststore) stored in a plaintext file readable by all (OS filesystem permissions set to 0440).
No default. Provide a keystore name and password when you create the private key and CSR for any host system.	For Java JDK, <code>cacerts</code> is the default unless the alternative default <code>jssecacerts</code> is available.

Keystore	Truststore
Must be owned by <code>hadoop</code> user and group so that HDFS, MapReduce, YARN can access the private key.	HDFS, MapReduce, and YARN need client access to truststore.

The details in the table above are specific to the Java KeyStore (JKS) format, which is used by Java-based cluster services such as Cloudera Manager Server, Cloudera Management Service, and many (but not all) CDH components and services. See [Certificate Formats \(JKS, PEM\) and Cluster Components](#) on page 195 for information about certificate and key file type used various processes.

CDH Services as TLS/SSL Servers and Clients

Cluster services function as a TLS/SSL server, client, or both:

Component	Client	Server
Flume	✓	✓
HBase	⊘	✓
HDFS	✓	✓
Hive	✓	✓
Hue (Hue is a TLS/SSL client of HDFS, MapReduce, YARN, HBase, and Oozie.)	✓	⊘
MapReduce	✓	✓
Oozie	⊘	✓
YARN	✓	✓

Daemons that function as TLS/SSL servers load the keystores when starting up. When a client connects to an TLS/SSL server daemon, the server transmits the certificate loaded at startup time to the client, and the client then uses its truststore to validate the certificate presented by the server.

Certificate Formats (JKS, PEM) and Cluster Components

Cloudera Manager Server, Cloudera Management Service, and many other CDH services use JKS formatted keystores and certificates. Cloudera Manager Agent, Hue, Key Trustee Server, Impala, and other Python or C++ based services require [PEM](#) formatted certificates and keystores rather than Java. Specifically, PEM certificates conform to PKCS #8, which requires individual Base64-encoded text files for certificate and password-protected private key file. The table summarizes certificate types required by several components.

Component	JKS	PEM
Flume	✓	⊘
HBase	✓	⊘
HDFS	✓	⊘
Hive (Hive clients and HiveServer 2)	✓	⊘
Hue	⊘	✓
Impala	⊘	✓

Component	JKS	PEM
MapReduce	✓	⊘
Oozie	✓	⊘
Solr	⊘	✓
YARN	✓	⊘

For more information, see:

- [How to Convert Certificate Encodings \(DER, JKS, PEM\) for TLS/SSL](#)
- [OpenSSL Cryptography and TLS/SSL Toolkit](#)

Recommended Keystore and Truststore Configuration

Cloudera recommends the following for keystores and truststores for Cloudera Manager clusters:

- Create a separate keystore for each host. Each keystore should have a name that helps identify it as to the type of host—server or agent, for example. The keystore contains the private key and should be password protected.
- Create a single truststore that can be used by the entire cluster. This truststore contains the root CA and intermediate CAs used to authenticate certificates presented during TLS/SSL handshake. The truststore does not need to be password protected. (See [How To Add Root and Intermediary CAs to Truststore](#) for more information about the truststore for TLS/SSL and Cloudera clusters.)

The steps included in [Generate TLS Certificates](#) follow this approach.

Step 2: Installing JCE Policy File for AES-256 Encryption



Note: This step is not required when using JDK 1.8.0_161 or greater. JDK 1.8.0_161 enables unlimited strength encryption by default.

By default, CentOS and Red Hat Enterprise Linux 5.5 (and higher) use AES-256 encryption for Kerberos tickets, so the [Java Cryptography Extension \(JCE\) Unlimited Strength Jurisdiction Policy File](#) must be installed on all cluster hosts as detailed below. Alternatively, the Kerberos instance can be [modified to not use AES-256](#).

To install the JCE Policy file on the host system at the OS layer:

1. Download the `jce_policy-x.zip`.
2. Unzip the file.
3. Follow the steps in the `README.txt` to install it.

To use Cloudera Manager to install the JCE policy file:

1. Log in to the Cloudera Manager Admin Console.
2. Enter the following URL in the browser:

```
http://<Cloudera_Manager_server_host>:7180/cm/upgrade-wizard/wizard
```

3. Select **Install Oracle Java SE Development Kit (JDK 8)**.
4. Select **Install Java Unlimited Strength Encryption Policy Files**.
5. Click **Continue**.

Alternative: Disable AES-256 encryption from the Kerberos instance:

1. Remove `aes256-cts:normal` from the `supported_enctypes` field of the `kdc.conf` or `krb5.conf` file.
2. Restart the Kerberos KDC and the kadmin server so the changes take effect.

The keys of relevant principals, such as Ticket Granting Ticket principal (`krbtgt/REALM@REALM`), might need to change.



Note: If AES-256 remains in use despite disabling it, it may be because the `aes256-cts:normal` setting existed when the Kerberos database was created. To resolve this issue, create a new Kerberos database and then restart both the KDC and the kadmin server.

To verify the type of encryption used in your cluster:

1. **For MIT KDC:** On the local KDC host, type this command in the `kadmin.local` or `kadmin` shell to create a test principal:

```
kadmin: addprinc test
```

For Active Directory: Create a new AD account with the name, `test`.

2. On a cluster host, type this command to start a Kerberos session as `test`:

```
kinit test
```

3. On a cluster host, type this command to view the encryption type in use:

```
klist -e
```

Step 2: Installing JCE Policy File for AES-256 Encryption

If AES is being used, output like the following is displayed after you type the `klist` command (note that AES-256 is included in the output):

```
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@Cloudera Manager
Valid starting      Expires            Service principal
05/19/11 13:25:04  05/20/11 13:25:04  krbtgt/Cloudera Manager@Cloudera Manager
    Etype (skey, tkt): AES-256 CTS mode with 96-bit SHA-1 HMAC, AES-256 CTS mode with
    96-bit SHA-1 HMAC
```

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.
```