1.Introduce yourself

Introduction , Education , work experience ( associate working from 3rd party company ) working with

2.day to day activity

Hadoop Admin Responsibilities:

* Responsible for implementation and ongoing administration of Hadoop infrastructure.
* Aligning with the systems engineering team to propose and deploy new hardware and software environments required for Hadoop and to expand existing environments.
* Working with data delivery teams to setup new Hadoop users. This job includes setting up Linux users, setting up Kerberos principals and testing HDFS, Hive, Pig and MapReduce access for the new users.
* Cluster maintenance as well as creation and removal of nodes using tools like Ganglia, Nagios, Cloudera Manager Enterprise, Dell Open Manage and other tools.
* Performance tuning of Hadoop clusters and Hadoop MapReduce routines.
* Screen Hadoop cluster job performances and capacity planning
* Monitor Hadoop cluster connectivity and security
* Manage and review Hadoop log files.
* File system management and monitoring.
* HDFS support and maintenance.
* Diligently teaming with the infrastructure, network, database, application and business intelligence teams to guarantee high data quality and availability.
* Collaborating with application teams to install operating system and Hadoop updates, patches, version upgrades when required.
* Point of Contact for Vendor escalation

Q 3. production cluster with configurations and services

**Overview of the Cluster Configuration Service**

The Pivotal GemFire cluster configuration service persists cluster configurations created by gfsh

commands to the locators in a cluster and distributes the configurations to members of the cluster

**Why Use the Cluster Configuration Service**

We highly recommend that you use the gfsh command line and the cluster configuration service as the primary mechanism to manage your cluster configuration. Specify configuration within a cache.xml file for only those items that cannot be specified or altered using gfsh. Using a common cluster configuration reduces the amount of time you spend configuring individual members and enforces consistent configurations when bringing up new members in your cluster. You no longer need to reconfigure each new member that you add to the cluster. You no longer need to worry about validating your cache.xml file. It also becomes easier to propagate configuration changes across your cluster and deploy your configuration changes to different environments.

You can use the cluster configuration service to:

* Save the configuration for an entire Pivotal GemFire cluster.
* Restart members using a previously-saved configuration.
* Export a configuration from a development environment and migrate that configuration to create a testing or production system.
* Start additional servers without having to configure each server separately.
* Configure some servers to host certain regions and other servers to host different regions, and configure all servers to host a set of common regions.

**Using the Cluster Configuration Service**

To use the cluster configuration service in GemFire, you must use dedicated, standalone locators in your deployment. You cannot use the cluster configuration service with co-located locators (locators running in another process such as a server) or in multicast environments.

The standalone locators distribute configuration to all locators in a cluster. Every locator in the cluster with --enable-cluster-configuration set to true keeps a record of all cluster-level and group-level configuration settings.

**Note:** The default behavior for gfsh is to create and save cluster configurations. You can disable the cluster configuration service by using the --enable-cluster-configuration=false option when starting locators.

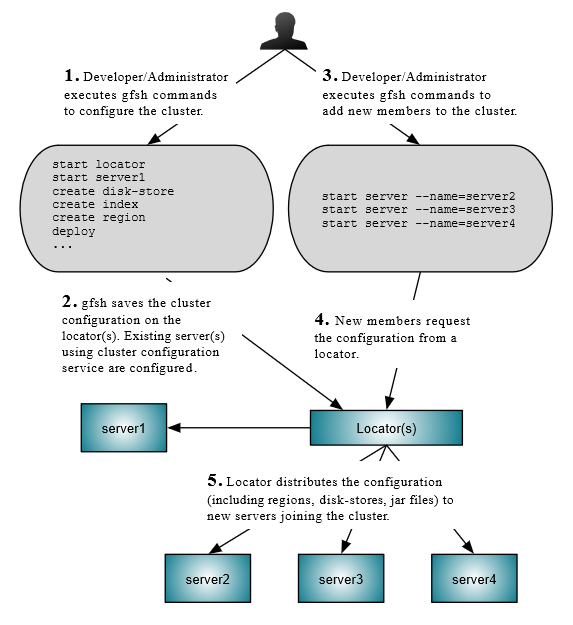
You can load existing configuration into the cluster by using the [gfsh import cluster-configuration](https://gemfire.docs.pivotal.io/98/geode/tools_modules/gfsh/command-pages/import.html#topic_vnv_grz_ck) command after starting up a locator.

Subsequently, any servers that you start with gfsh that have --use-cluster-configuration set to true will pick up the cluster configuration from the locator as well as any appropriate group-level configurations (for member groups they belong to). To disable the cluster configuration service on a server, you must start the server with the --use-cluster-configuration parameter set to false. By default, the parameter is set to true.

**How the Cluster Configuration Service Works**

When you use gfsh commands to create Pivotal GemFire regions, disk-stores, and other objects, the cluster configuration service saves the configurations on each locator in the cluster. If you specify a group when issuing these commands, a separate configuration is saved containing only configurations that apply to the group.

When you use gfsh to start new Pivotal GemFire servers, the locator distributes the persisted configurations to the new server. If you specify a group when starting the server, the server receives the group-level configuration in addition to the cluster-level configuration. Group-level configurations are applied after cluster-wide configurations; therefore you can use group-level to override cluster-level settings.



**gfsh Commands that Create Cluster Configurations**

The following gfsh commands cause the configuration to be written to all locators in the cluster (the locators write the configuration to disk):

* configure pdx\*
* create region
* alter region
* alter runtime
* destroy region
* create index
* destroy index
* create disk-store
* destroy disk-store
* create async-event-queue
* deploy jar
* undeploy jar
* create gateway-sender
* create gateway-receiver

**\*** Note that the configure pdx command must be executed *before* starting your data members. This command does not affect any currently running members in the system. Data members (with cluster configuration enabled) that are started after running this command will pick up the new PDX configuration.

**gfsh Limitations**

These are the configurations that you cannot create or alter using gfsh. These configurations must be within a cache.xml file or be applied by using the API:

* Client cache configuration
* You cannot directly modify the attributes of the following objects:
  + function
  + custom-load-probe
  + compressor
  + serializer
  + instantiator
  + pdx-serializer

**Note:** The configure pdx command always specifies the org.apache.geode.pdx.Reflection Based AutoSerializer class. You cannot specify a custom PDX serializer in gfsh.

* + initializer
  + lru-heap-percentage
  + lru-memory-size
  + partition-resolver
  + partition-listener
  + transaction-listener
  + transaction-writer
* Adding or removing a TransactionListener
* Deleting an AsyncEventQueue
* Configuring a GatewayConflictResolver
* You cannot specify parameters and values for Java classes for the following:
  + gateway-listener
  + gateway-conflict-resolver
  + gateway-event-filter
  + gateway-transport-filter
  + gateway-event-substitution-filter

**Disabling the Cluster Configuration Service**

If you do not want to use the cluster configuration service, start up your locator with the --enable-cluster-configuration parameter set to false or do not use standalone locators. You will then need to configure the cache (via cache.xml or API) separately on all your cluster members.

Q 4. which version of CDH and CM you are using

* Cloudera Manager 4 and CDH 4 have reached End of Maintenance (EOM) on August 9, 2015. Cloudera does not support or provide updates for Cloudera Manager 4 and CDH 4 releases.
* Cloudera Manager 3 and CDH 3 have reached End of Maintenance (EOM) on June 20, 2013. Cloudera does not support or provide updates for Cloudera Manager 3 and CDH 3 releases.

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:

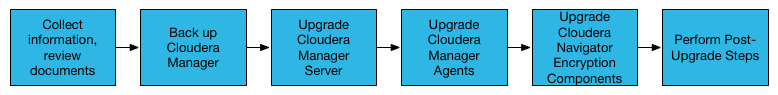
* [Upgrade Overview](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_overview.html#upgrade-overview)
* [Upgrading Cloudera Manager](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_cm_upgrade.html)
* [Upgrading CDH](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_cdh_upgrading_top.html)

Procedures for upgrading **Cloudera Director** are discussed in the Cloudera Director documentation. See [Upgrading Cloudera Director](https://www.cloudera.com/documentation/director/latest/topics/director_upgrade.html).

Upgrade Overview

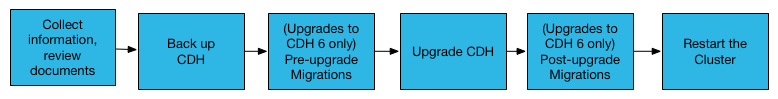
Upgrading consists of two major steps, upgrading Cloudera Manager and upgrading a CDH cluster.

***Upgrading Cloudera Manager***



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.

***Upgrading a CDH Cluster***



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: [Show](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_overview.html)

**Note:**

This 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.

Continue reading:

* [Assessing the Impact of an Upgrade](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_overview.html#concept_ch1_xw1_yw)
* [Overview of Upgrading Cloudera Manager](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_overview.html#concept_xkm_f4q_tw)
* [Overview of Upgrading CDH](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_overview.html#concept_vpj_f4q_tw)
* [Overview of Upgrading Cloudera Navigator Components](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_overview.html#concept_fb4_y3m_zw)

**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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/rg_release_notes.html#enterprise_release_guide)
* [Cloudera Enterprise 5.x Release Notes](https://www.cloudera.com/documentation/enterprise/release-notes/topics/rg_release_notes.html)

Also check the [Cloudera Enterprise 6 Requirements and Supported Versions](https://www.cloudera.com/documentation/enterprise/upgrade/topics/rg_requirements_supported_versions.html#c6_requirements) 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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_upgrade_paths.html#ug_upgrade_paths). **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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/cm_mc_client_config.html#cmug_topic_5_9) 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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/cm_mc_client_config.html#cmug_topic_5_9) are redeployed.

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

**Maintenance Upgrades**

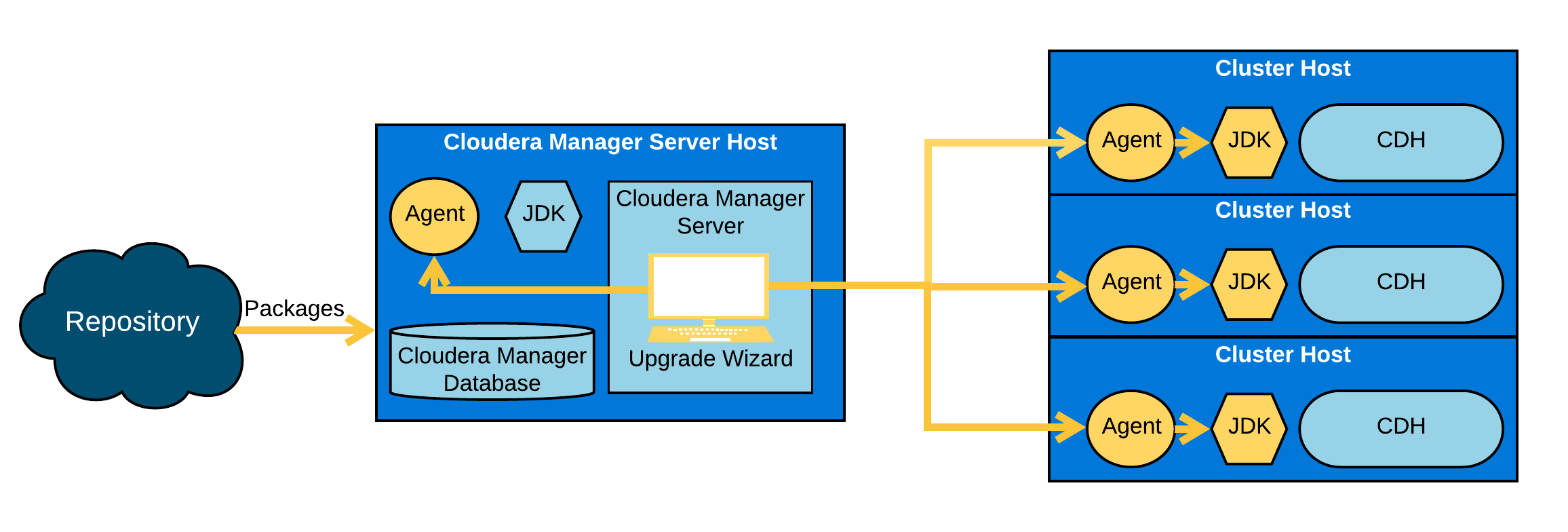
Maintenance upgrades fix critical bugs or address security issues. No new functionality or incompatible changes are introduced. The version numbers for maintenance releases differ only in the third digit, for example, when upgrading from version 6.0.0 to 6.0.1.

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**

***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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_cm_upgrade.html).

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](https://www.cloudera.com/documentation/enterprise/5/latest/topics/cm_ag_ug_cm5_tarballs.html) for procedures to upgrade to the latest version of Cloudera Manager 5.x. (Click **Other Versions** to locate the document for earlier versions.)

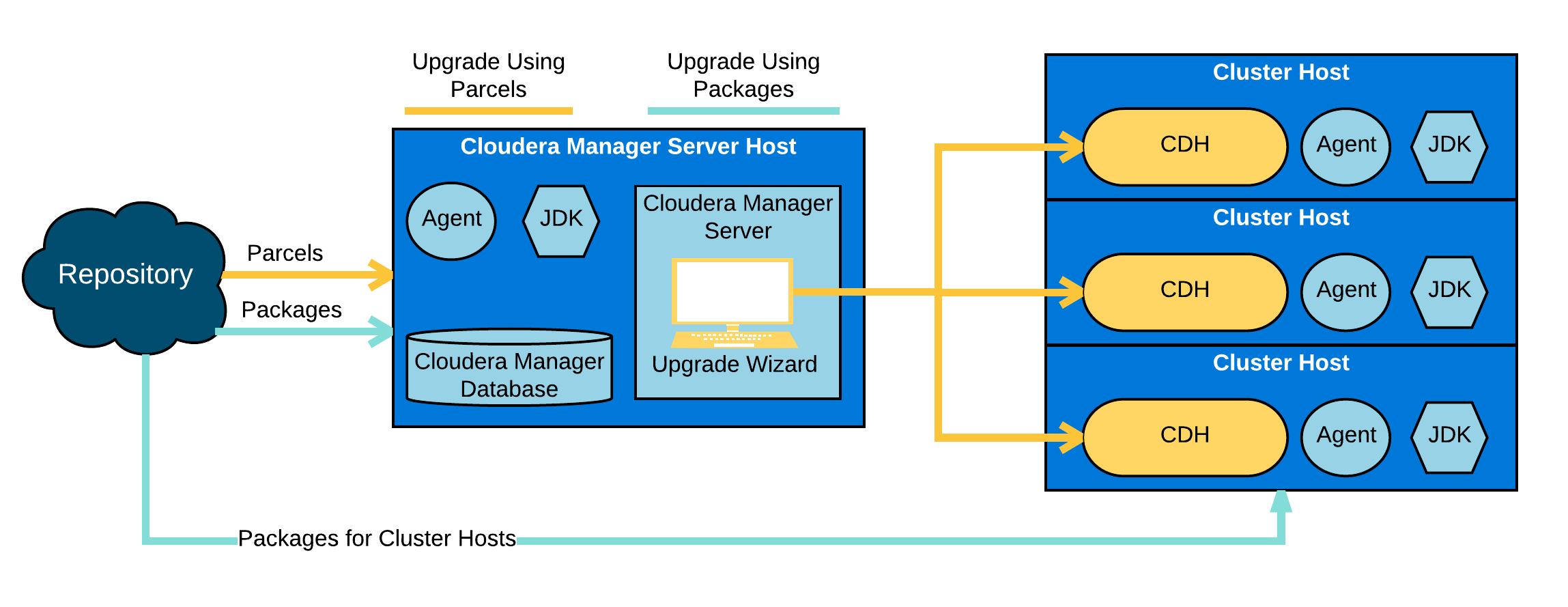
**Overview of Upgrading CDH**

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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_overview.html#concept_ch1_xw1_yw). 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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/cm_ig_managing_software.html#xd_583c10bfdbd326ba--6eed2fb8-14349d04bee--772f).

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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/cdh_hag_hdfs_ha_config.html#topic_2), 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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/ug_cdh_upgrade.html#cdh_upgrade5).

***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](https://www.cloudera.com/documentation/enterprise/upgrade/topics/cm_ig_migrating_packages_to_parcels.html#migrate_pkg_to_parcel) 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)](https://www.cloudera.com/documentation/enterprise/upgrade/topics/rg_pcm_navigator_encryption.html#pcm_navigator_encryption) or [Product Compatibility Matrix for Cloudera Navigator Encryption (Cloudera Manager 5.x)](https://www.cloudera.com/documentation/enterprise/release-notes/topics/rn_consolidated_pcm.html#pcm_navigator_encryption).

Q 5. Do you have Hbase in your cluster

Deploying HBase on a Cluster

After you have HBase running in pseudo-distributed mode, the same configuration can be extended to running on a distributed cluster.

**Note:** **Before you start**

This section assumes that you have already installed the [HBase Master](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_hbase_standalone_start.html#topic_20_4_1) and the [HBase RegionServer](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_hbase_pseudo_configure.html#topic_20_5_3__title_607) and gone through the steps for [standalone](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_hbase_standalone_start.html#topic_20_4) and [pseudo-distributed](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_hbase_pseudo_configure.html#topic_20_5) configuration. You are now about to distribute the processes across multiple hosts; see [Choosing Where to Deploy the Processes](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_hbase_cluster_deploy.html#topic_20_6_1).

Choosing Where to Deploy the Processes

For small clusters, Cloudera recommends designating one node in your cluster as the HBase Master node. On this node, you will typically run the HBase Master and a ZooKeeper quorum peer. These master processes may be collocated with the Hadoop NameNode and JobTracker for small clusters.

Designate the remaining nodes as RegionServer nodes. On each node, Cloudera recommends running a RegionServer, which may be collocated with a Hadoop TaskTracker (MRv1) and a DataNode. When co-locating with TaskTrackers, be sure that the resources of the machine are not oversubscribed – it's safest to start with a small number of MapReduce slots and work up slowly.

The HBase Thrift service is light-weight, and can be run on any node in the cluster.

Configuring for Distributed Operation

After you have decided which machines will run each process, you can edit the configuration so that the nodes can locate each other. In order to do so, you should make sure that the configuration files are synchronized across the cluster. Cloudera strongly recommends the use of a configuration management system to synchronize the configuration files, though you can use a simpler solution such as rsync to get started quickly.

The only configuration change necessary to move from pseudo-distributed operation to fully-distributed operation is the addition of the ZooKeeper Quorum address in hbase-site.xml. Insert the following XML property to configure the nodes with the address of the node where the ZooKeeper quorum peer is running:

<property>

<name>hbase.zookeeper.quorum</name>

<value>mymasternode</value>

</property>

The hbase.zookeeper.quorum property is a comma-separated list of hosts on which ZooKeeper servers are running. If one of the ZooKeeper servers is down, HBase will use another from the list. By default, the ZooKeeper service is bound to port 2181. To change the port, add the hbase.zookeeper.property.clientPort property to hbase-site.xml and set the value to the port you want ZooKeeper to use. In CDH 5.7.0 and higher, you do not need to use hbase.zookeeper.property.clientPort. Instead, you can specify the port along with the hostname for each ZooKeeper host:

<property>

<name>hbase.zookeeper.quorum</name>

<value>zk1.example.com:2181,zk2.example.com:20000,zk3.example.com:31111</value>

</property>

For more information, see [this chapter](http://hbase.apache.org/book/zookeeper.html) of the Apache HBase Reference Guide.

To start the cluster, start the services in the following order:

1. The ZooKeeper Quorum Peer
2. The HBase Master
3. Each of the HBase RegionServers

After the cluster is fully started, you can view the HBase Master web interface on port 60010 and verify that each of the RegionServer nodes has registered properly with the master.

See also [Accessing HBase by using the HBase Shell](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_hbase_shell.html#topic_20_7), [Using MapReduce with HBase](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_mapreduce_hbase.html#topic_20_8) and [Troubleshooting HBase](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_ig_hbase_troubleshooting.html#topic_20_9). For instructions on improving the performance of local reads, see [Optimizing Performance in CDH](https://www.cloudera.com/documentation/enterprise/5-7-x/topics/cdh_admin_performance.html).

Q 6. Do you have kerberos configured in your cluster

# How to Configure Clusters to Use Kerberos for Authentication

Cloudera clusters can use Kerberos to authenticate services running on the cluster and the users who need access to those services. This How To guide provides the requirements, pre-requisites, and high-level summary of the steps needed to integrate clusters with Kerberos for authentication.

Note: For clusters deployed using Cloudera Manager Server, Cloudera recommends using the Kerberos configuration wizard available through the Cloudera Manager Admin Console. See [Enabling Kerberos Authentication Using the Wizard](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_authentication.html#xd_583c10bfdbd326ba--6eed2fb8-14349d04bee--766e) for details.

The following are the general steps for integrating Kerberos with Cloudera clusters without using the Cloudera Manager configuration wizard.

Continue reading:

* [Step 1: Verify Requirements and Assumptions](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#requirements-assumptions)
* [Step 2. Create Principal for Cloudera Manager Server in the Kerberos KDC](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#create-principal-for-cm)
* [Step 3: Add the Credentials for the Principal to the Cluster](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#cmac-import-from-kdc)
* [Step 4: Identify Default Kerberos Realm for the Cluster](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#cfg-default-krb-realm)
* [Step 5: Stop all Services](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#stop-services)
* [Step 6. Specify Kerberos for Security](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#enable-kerberos-cluster)
* [Step 7: Restart All Services](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#start-all-svcs)
* [Step 8: Deploy Client Configurations](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#deploy-client-cfgs)
* [Step 9: Create the HDFS Superuser Principal](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#create-hdfs-superuser)
* [Step 11: Prepare the Cluster for Each User](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#prep-cluster-for-users)
* [Step 12: Verify Successful Kerberos Integration](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#verify-successful-kerberos-integration)

## Step 1: Verify Requirements and Assumptions

The steps outlined below assume that:

* The Kerberos instance has been setup, is running, and is available during the configuration process.
* The Cloudera cluster has been installed and is operational, with all services fully-functional—Cloudera Manager Server, CDH, and Cloudera Manager Agent processes on all cluster nodes.

### Hosts Configured for AES-256 Encryption

By default, CentOS and RHEL 6 (and higher) use AES-256 encryption for Kerberos tickets. If you use either of these platforms for your cluster, the [Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy File](http://www.oracle.com/technetwork/java/javase/downloads/index.html) must be installed on all cluster hosts.

**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.

Note: The AES-256 encryption can also be configured on a running cluster by using Cloudera Manager Admin Console. See [To use Cloudera Manager to install the JCE policy file](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_s2_jce_policy.html#xd_583c10bfdbd326ba--6eed2fb8-14349d04bee--7737__cm-jce-policy) for details.

### Required Administrator Privileges

Setting up the Cloudera cluster to use Kerberos for authentication requires complete administrator access to the cluster and administrator privileges on the Kerberos instance:

* [Cluster Administrator](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_user_roles.html#concept_wfh_tvy_qp) or [Full Administrator](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_user_roles.html#concept_wfh_tvy_qp)
* Kerberos administrator privileges:

someone/admin@YOUR-DOMAIN.FQDN.EXAMPLE.COM

If you do not have administrator privileges on the Kerberos instance, you will need help from the Kerberos administrator before you can complete the process.

### Required User (Service Account) Directories

During installation, the cloudera-scm account is created on the host system. When Cloudera Manager and CDH services are installed at the same time, Cloudera Manager creates other accounts as needed to support the service role daemons. However, if the CDH services and Cloudera Manager are installed separately, you may need to specifically set directory permissions for certain Hadoop user (service daemon) accounts for successful integration with Kerberos. The following table shows the accounts used for core service roles. Note that hdfs acts as superuser for the system.

| **User** | **Service Roles** |
| --- | --- |
| Hdfs | NameNode, DataNodes, Secondary NameNode (and HDFS superuser) |
| mapred | JobTracker, TaskTrackers (MR1), Job History Server (YARN) |
| Yarn | ResourceManager, NodeManager (YARN) |
| Oozie | Oozie Server |
| Hue | Hue Server, Beeswax Server, Authorization Manager, Job Designer |

These accounts require ownership control over specific directories.

* For newly installed Cloudera clusters (Cloudera Manager and CDH installed at the same time)—The Cloudera Manager Agent process on each cluster host automatically configures the appropriate directory ownership when the cluster launches.
* For existing CDH clusters using HDFS and running MapReduce jobs prior to Cloudera Manager installation—The directory ownership must be manually configured, as shown in the table below. The directory owners cannot differ from those shown in the table to ensure that the service daemons can set permissions as needed on each directory.

| **Directory Specified in this Property** | **Owner** |
| --- | --- |
| dfs.name.dir | hdfs:hadoop |
| dfs.data.dir | hdfs:hadoop |
| mapred.local.dir | mapred:hadoop |
| mapred.system.dir in HDFS | mapred:hadoop |
| yarn.nodemanager.local-dirs | yarn:yarn |
| yarn.nodemanager.log-dirs | yarn:yarn |
| oozie.service.StoreService.jdbc.url (if using Derby) | oozie:oozie |
| [[database]] name | hue:hue |
| javax.jdo.option.ConnectionURL | hue:hue |

## Step 2. Create Principal for Cloudera Manager Server in the Kerberos KDC

Cloudera Manager Server has its own principal to connect to the Kerberos KDC and import user and service principals for use by the cluster.

The steps below summarize the process of adding a principal specifically for Cloudera Manager Server to an MIT KDC and an Active Directory KDC. See documentation from MIT, Microsoft, or the appropriate vendor for more detailed information.

Note: If an administrator principal to act on behalf of Cloudera Manager cannot be created on the Kerberos KDC for whatever reason, Cloudera Manager will not be able to create or manage principals and keytabs for CDH services. That means these principals must be created manually on the Kerberos KDC and then imported (retrieved) by Cloudera Manager. See [Using a Custom Kerberos Keytab Retrieval Script](https://www.cloudera.com/documentation/enterprise/latest/topics/sg_keytab_retrieval_script.html#xd_583c10bfdbd326ba--69adf108-1492ec0ce48--7c0f) for details about this process.

### Creating a Principal in Active Directory

Check your Microsoft documentation for specific details for your Active Directory KDC. The general process is as follows:

1. Create an Organizational Unit (OU) in your Active Directory KDC service that will contain the principals for use by the CDH cluster.
2. Add a new user account to Active Directory, for example, username@YOUR-REALM.EXAMPLE.COM. Set the password for the user to never expire.
3. Use the Delegate Control wizard of Active Directory and grant this new user permission to Create, Delete, and Manage User Accounts.

### Creating a Principal in an MIT KDC

For MIT Kerberos, user principals that include the instance name admin designate a user account with administrator privileges. For example:

username/admin@YOUR-REALM.EXAMPLE.COM

Create the Cloudera Manager Server principal as shown in one of the examples below, appropriate for the location of the Kerberos instance and using the correct REALM name for your setup.

**For MIT Kerberos KDC on a remote host:**

kadmin: addprinc -pw password cloudera-scm/admin@YOUR-REALM.EXAMPLE.COM

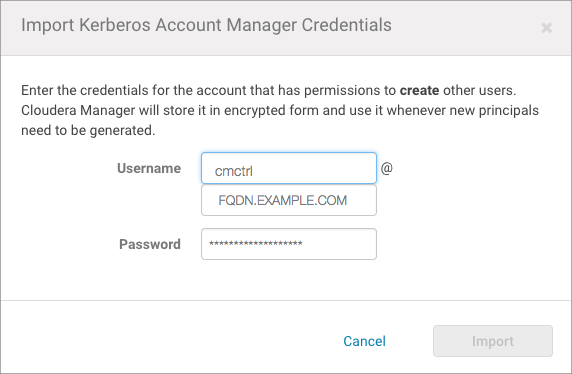
**For MIT Kerberos KDC on a local host:**

kadmin.local: addprinc -pw password cloudera-scm/admin@YOUR-REALM.EXAMPLE.COM

## Step 3: Add the Credentials for the Principal to the Cluster

Assuming the principal was successfully added to the Kerberos KDC, it can be added to the cluster as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select Administration > Security.
3. Click the Kerberos Credentials tab.
4. Click the Import Kerberos Account Manager Credentials button.
5. Enter the credentials for the principal added to the [Kerberos KDC in the previous step](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#create-principal-for-cm):
   * For Username, enter the primary and realm portions of the Kerberos principal. Enter the realm name in all upper-case only (YOUR-REALM.EXAMPLE.COM) as shown below.
   * Enter the Password for the principal.



1. Click Import.

Cloudera Manager encrypts the username and password into a keytab and uses it to create new principals in the KDC as needed.

Click Close when complete.

## Step 4: Identify Default Kerberos Realm for the Cluster

Each host in the cluster must have the default realm property (default\_realm) specified in the libdefaults section of its Kerberos configuration file (/etc/krb5.conf).

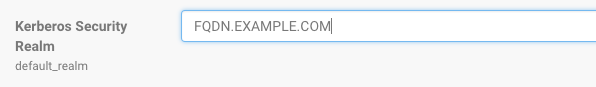
[libdefaults]

default\_realm = FQDN.EXAMPLE.COM

After adding the default realm to the configuration file for all hosts in the cluster, configure the same default realm for Cloudera Manager Server.

In the Cloudera Manager Admin Console:

1. Select Administration > Settings.
2. Select Kerberos for the Category filter.
3. In the Kerberos Security Realm field, enter the default realm name set in the Kerberos configuration file (/etc/krb5.conf) of each host in the cluster. For example:



1. Click **Save Changes**.

## Step 5: Stop all Services

All service daemons in the cluster must be stopped so that they can be restarted at the same time and start as authenticated services in the cluster. Service daemons running without authenticating to Kerberos first will not be able to communicate with other daemons in the cluster that have authenticated to Kerberos, so they must be shut down and restarted at the end of the configuration process, as a unit.

Note: The requirement to stop all daemons prevents using the rolling upgrade process to enable Kerberos integration on the cluster.

Stop all running services and the Cloudera Management Service as follows:

In the Cloudera Manager Admin Console:

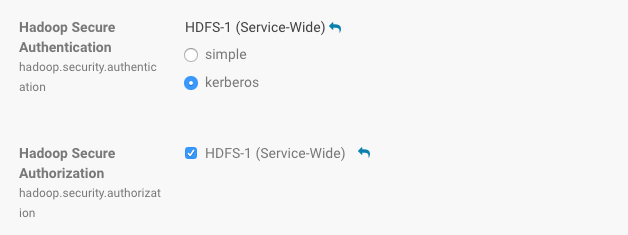
1. Select Clusters > Cluster-n.
2. Click the Actions drop-down menu and select Stop to stop all services on the cluster.
3. Click Stop on the warning message to stop all services on the cluster. The Command Details window displays the progress as each service shuts down. When the message All services successfully stopped displays, close the Command Details window.
4. Select > Clusters > Cloudera Management Service.
5. Click the Actions drop-down menu and select Stop to stop the Cloudera Management Service. The Command Details window displays the progress as each role instance running on the Cloudera Management Service shuts down. The process is completed when the message Command completed with n/n successful subcommands displays.
6. Click Close.

## Step 6. Specify Kerberos for Security

Kerberos must be specified as the security mechanism for Hadoop infrastructure, starting with the HDFS service. Enable Cloudera Manager Server security for the cluster on an HDFS service. After you do so, the Cloudera Manager Server automatically enables Hadoop security on the MapReduce and YARN services associated with that HDFS service.

In the Cloudera Manager Admin Console:

1. Select Clusters > HDFS-n.
2. Click the Configuration tab.
3. Select HDFS-n for the Scope filter.
4. Select Security for the Category filter.
5. Scroll (or search) to find the Hadoop Secure Authentication property.
6. Click the kerberos button to select Kerberos:



1. Click the value for the Hadoop Secure Authorization property and select the checkbox to enable service-level authorization on the selected HDFS service. You can specify comma-separated lists of users and groups authorized to use Hadoop services or perform admin operations using the following properties under the Service-Wide Security section:
   * Authorized Users: Comma-separated list of users authorized to use Hadoop services.
   * Authorized Groups: Comma-separated list of groups authorized to use Hadoop services.
   * Authorized Admin Users: Comma-separated list of users authorized to perform admin operations on Hadoop.
   * Authorized Admin Groups: Comma-separated list of groups authorized to perform admin operations on Hadoop.

Important: For Cloudera Manager's Monitoring services to work, the hue user should always be added as an authorized user.

1. In the Search field, type DataNode Transceiver to find the DataNode Transceiver Port property.
2. Click the value for the DataNode Transceiver Port property and specify a privileged port number (below 1024). Cloudera recommends 1004. Note: If there is more than one DataNode Role Group, you must specify a privileged port number for each DataNode Transceiver Port property.
3. In the Search field, type DataNode HTTP to find the DataNode HTTP Web UI Port property and specify a privileged port number (below 1024). Cloudera recommends 1006. Note: The port numbers for the two DataNode properties must be below 1024 to provide part of the security mechanism to make it impossible for a user to run a MapReduce task that impersonates a DataNode. The port numbers for the NameNode and Secondary NameNode can be anything you want, but the default port numbers are good ones to use.
4. In the Search field type Data Directory Permissions to find the DataNode Data Directory Permissions property.
5. Reset the value for the DataNode Data Directory Permissions property to the default value of 700 if not already set to that.
6. Make sure you have changed the DataNode Transceiver Port, DataNode Data Directory Permissions and DataNode HTTP Web UI Port properties for every DataNode role group.
7. Click Save Changes to save the configuration settings.

To enable ZooKeeper security:

1. Go to the ZooKeeper Service Configuration tab and click View and Edit.
2. Click the value for Enable Kerberos Authentication property.
3. Click Save Changes to save the configuration settings.

To enable HBase security:

1. Go to the HBase Service > Configuration tab and click View and Edit.
2. In the Search field, type HBase Secure to show the Hadoop security properties (found under the Service-Wide > Security category).
3. Click the value for the HBase Secure Authorization property and select the checkbox to enable authorization on the selected HBase service.
4. Click the value for the HBase Secure Authentication property and select kerberos to enable authorization on the selected HBase service.
5. Click Save Changes to save the configuration settings.

To enable Solr security:

1. Go to the Solr Service > Configuration tab and click View and Edit.
2. In the Search field, type Solr Secure to show the Solr security properties (found under the Service-WideSecurity category).
3. Click the value for the Solr Secure Authentication property and select kerberos to enable authorization on the selected Solr service.
4. Click Save Changes to save the configuration settings.

Note: Using Cloudera Manager Admin Console to generate client configuration files **after** enabling Kerberos does not provide the Kerberos principals and keytabs that users need to authenticate to the cluster. Users must obtain their Kerberos principals from the Kerberos administrator and then run the kinit command themselves.

### Credentials Generated

After you enable security for any of the services in Cloudera Manager, a command called Generate Credentials will be triggered automatically. You can watch the progress of the command on the top right corner of the screen that shows the running commands. Wait for this command to finish (indicated by a grey box containing "0" in it).

## Step 7: Restart All Services

Start all services on the cluster using the Cloudera Manager Admin Console:

1. Select Clusters > Cluster-n.
2. Click the Actions drop-down button menu and select Start. The confirmation prompt displays.
3. Click Start to confirm and continue. The Command Details window displays progress. When All services successfully started displays, close the Command Detailswindow.
4. Select > Clusters > Cloudera Management Service.
5. Click the Actions drop-down menu and select Start. The Command Details window displays the progress as each role instance running on the Cloudera Management Service starts up. The process is completed when the message Command completed with n/n successful subcommands displays.

## Step 8: Deploy Client Configurations

Deploy client configurations for services supported on the cluster using the Cloudera Manager Admin Console:

1. Select Clusters > Cluster-n.
2. Click the Actions drop-down button menu and select Deploy Client Configuration.

## Step 9: Create the HDFS Superuser Principal

To be able to create home directories for users, you will need access to the HDFS superuser account. (CDH automatically created the HDFS superuser account on each cluster host during CDH installation.) When you enabled Kerberos for the HDFS service, you lost access to the default HDFS superuser account using sudo -u hdfs commands. Cloudera recommends you use a different user account as the superuser, not the default hdfs account.

### Designating a Non-Default Superuser Group

To designate a different group of superusers instead of using the default hdfs account, follow these steps:

1. Go to the Cloudera Manager Admin Console and navigate to the HDFS service.
2. Click the Configuration tab.
3. Select Scope > HDFS (Service-Wide).
4. Select Category > Security.
5. Locate the **Superuser Group** property and change the value to the appropriate group name for your environment. For example, superuser.
6. Click Save Changes.
7. Restart the HDFS service.

To enable your access to the superuser account now that Kerberos is enabled, you must now create a Kerberos principal or an Active Directory user whose first component is superuser:

#### For Active Directory

Add a new user account to Active Directory, superuser@YOUR-REALM.EXAMPLE.COM . The password for this account should be set to never expire.

#### For MIT KDC

1. In the kadmin.local or kadmin shell, type the following command to create a Kerberos principal called superuser:

kadmin: addprinc superuser@YOUR-REALM.EXAMPLE.COM

This command prompts you to create a password for the superuser principal. You should use a strong password because having access to this principal provides superuser access to all of the files in HDFS.

1. To run commands as the HDFS superuser, you must obtain Kerberos credentials for the superuser principal. To do so, run the following command and provide the appropriate password when prompted.

kinit superuser@YOUR-REALM.EXAMPLE.COM

### Step 10: Get or Create a Kerberos Principal for Each User Account

Now that Kerberos is configured and enabled on your cluster, you and every other Hadoop user must have a Kerberos principal or keytab to obtain Kerberos credentials to be allowed to access the cluster and use the Hadoop services. In the next step of this procedure, you need to create your own Kerberos principals to verify that Kerberos security is working on your cluster. If you and the other Hadoop users already have a Kerberos principal or keytab, or if your Kerberos administrator can provide them, you can skip ahead to the next step.

To create Kerberos principals for all users:

#### Active Directory

Add a new AD user account, <username>@EXAMPLE.COM for each Cloudera Manager service that should use Kerberos authentication. The password for these service accounts should be set to never expire.

##### MIT KDC

1. In the kadmin.local or kadmin shell, use the following command to create a principal for your account by replacing EXAMPLE.COM with the name of your realm, and replacing username with a username:

kadmin: addprinc username@EXAMPLE.COM

1. When prompted, enter the password twice.

## Step 11: Prepare the Cluster for Each User

Before you and other business users can access the cluster, the hosts must be prepared for each user. Perform the following tasks to prepare the hosts for each user:

1. Each host in the cluster must have a Unix user account with the same name as primary component of the user's principal name. For example, the principal jcarlos@YOUR-REALM.EXAMPLE.COM needs the Linux account jcarlos on each host system. Use LDAP (OpenLDAP, Microsoft Active Directory) for this step if possible. Note: Each account must have a user ID that is greater than or equal to 1000. In the /etc/hadoop/conf/taskcontroller.cfg file, the default setting for the banned.users property is mapred, hdfs, and bin to prevent jobs from being submitted from those user accounts. The default setting for the min.user.id property is 1000 to prevent jobs from being submitted with a user ID less than 1000, which are conventionally Unix super users.
2. Create a subdirectory under /user on HDFS for each user account (for example, /user/jcarlos). Change the owner and group of that directory to be the user.
3. hadoop fs -mkdir /user/jcarlos

hadoop fs -chown jcarlos /user/jcarlos

Note: The commands above do not include sudo -u hdfs because it is not required with Kerberos configured for the cluster (assuming you created the Kerberos credentials for the HDFS super user as detailed in [Step 9: Create the HDFS Superuser Principal](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#create-hdfs-superuser)).

## Step 12: Verify Successful Kerberos Integration

To verify that Kerberos has been successfully integrated for the cluster, try running one of the sample MapReduce jobs (sleep or pi, for example) provided at:

/usr/lib/hadoop/hadoop-examples.jar

This assumes you have the fully-functional cluster as recommended in [Step 1: Verify Requirements and Assumptions](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#requirements-assumptions) and that the client configure files have been generated as detailed in [Step 8: Deploying Client Configuration Files](https://www.cloudera.com/documentation/enterprise/latest/topics/cm_sg_using_cm_sec_config.html#deploy-client-cfgs).

**To verify that Kerberos security is working:**

1. Obtain Kerberos credentials for your user account.
2. $ kinit

youruserid@YOUR-REALM.EXAMPLE.COM

1. Enter a password when prompted.
2. Submit a sample pi calculation as a test MapReduce job. Use the following command if you use a package-based setup for Cloudera Manager:
3. $ hadoop jar /usr/lib/hadoop-0.20/hadoop-0.20.2\*examples.jar pi 10 10000
4. Number of Maps = 10
5. Samples per Map = 10000
6. ...
7. Job Finished in 38.572 seconds

Estimated value of Pi is 3.14120000000000000000

If you have a parcel-based setup, use the following command instead:

$ hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar pi 10 10000

Number of Maps = 10

Samples per Map = 10000

...

Job Finished in 30.958 seconds

Estimated value of Pi is 3.14120000000000000000

You have now verified that Kerberos security is working on your cluster.

Important:

Running a MapReduce job will fail if you do not have a valid Kerberos ticket in your credentials cache. You can examine the Kerberos tickets currently in your credentials cache by running the klist command. You can obtain a ticket by running the kinit command and either specifying a keytab file containing credentials, or entering the password for your principal. If you do not have a valid ticket, you will receive an error such as:

11/01/04 12:08:12 WARN ipc.Client:

Exception encountered while connecting to the server :

javax.security.sasl.SaslException:GSS initiate failed

[Caused by GSSException: No valid credentials provided (Mechanism level: Failed to find any

Kerberos tgt)]

Bad connection to FS. command aborted. exception: Call to nn-host/10.0.0.2:8020 failed on local exception:

java.io.IOException:javax.security.sasl.SaslException: GSS initiate failed

[Caused by GSSException: No valid credentials provided

(Mechanism level: Failed to find any Kerberos tgt)]

Q 7. what type of project you currently working on : - ask to sir

Q 8. what type of data you working on :- ask to sir

Q 9.how do you import that data :-

Your company may use CRM,ERP tools. But we don't exactly know how the data is organized & structured.

If we leave simple HDFS commands like put, copyFromLocal etc to load data into HDFS compatible format, below are the main tools to load data into HDFS

[**Apache Sqoop(TM)**](http://sqoop.apache.org/) is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured datastores such as relational databases. Data from MySQL, SQL Server & Oracle tables can be loaded into HDFS with this tool.

[**Apache Flume**](http://flume.apache.org/) is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data. It has a simple and flexible architecture based on streaming data flows. It is robust and fault tolerant with tunable reliability mechanisms and many failover and recovery mechanisms.

[**Other tools**](http://thinkbig.teradata.com/leading_big_data_technologies/ingestion-and-streaming-with-storm-kafka-flume/) include **Chukwa,Storm and Kafka**

But other important technology, which is becoming very popular is Spark. It is a Friend & Foe for Hadoop.

[**Spark**](http://spark.apache.org/) is emerging an good alternative to Hadoop for real time data processing, which may or may not use HDFS as data source.

Q 10 Is it by using streaming or batch processing

Real-Time Operating Systems

Real-time operating systems typically refer to the *reactions* to data. A system can be categorized as real-time if it can guarantee that the reaction will be within a tight real-world deadline, usually in a matter of seconds or milliseconds.

One of the best examples of a real-time system are those used in the stock market. If a stock quote should come from the network within 10 milliseconds of being placed, this would be considered a real-time process. Whether this was achieved by using a software architecture that utilized stream processing or just processing in hardware is irrelevant; the guarantee of the tight deadline is what makes it real-time. Some other samples of when using real-time systems would be beneficial are bank ATMs, air traffic control, and anti-lock braking systems.

Challenges

While this type of system sounds like a game changer, the reality is that real-time systems can be extremely hard to implement through the use of common software systems. As these systems take control over the program execution, it brings an entirely new level of abstraction. What this means is that the distinction between the control-flow of your program and the source code is no longer apparent because the real-time system chooses which task to execute at that moment. This is beneficial, as it allows for higher productivity using higher abstraction and can make it easier to design complex systems, but it means less control overall, which can be difficult to debug and validate.

Another common challenge with real-time operating systems is that the tasks are not isolated entities. The system decides which to schedule and sends out higher priority tasks before lower priority ones, thereby delaying their execution until all the higher priority tasks are completed.

More and more, some software systems are starting to go for a flavor of real-time processing where the deadline is not such an absolute as it is a probability. Known as soft real-time systems, they are able to usually or generally meet their deadline, although performance will begin to degrade if too many deadlines are missed.

Batch Processing

Batch processing is the processing of a large volume of data all at once. The data easily consists of millions of records for a day and can be stored in a variety of ways (file, record, etc). The jobs are typically completed simultaneously in non-stop, sequential order. An example of a batch processing job is all of the transactions a financial firm might submit over the course of a week. It can also be used in payroll processes, line item invoices, and supply chain and fulfillment.

Batch data processing is an extremely efficient way to process large amounts of data that is collected over a period of time. It also helps to reduce the operational costs that businesses might spend on labor as it doesn’t require specialized data entry clerks to support its functioning. It can be used offline and gives managers complete control as to when to start the processing, whether it be overnight or at the end of a week or pay period.

Challenges

As with anything, there are a few disadvantages to utilizing batch processing software. One of the biggest issues that businesses see is that debugging these systems can be tricky. If you don’t have a dedicated IT team or professional, trying to fix the system when an error occurs could be detrimental, causing the need for an outside consultant to assist.

Another problem with batch processing is that companies usually implement it to save money, but the software and training requires a decent amount of expenses in the beginning. Managers will need to be trained to understand how to schedule a batch, what triggers them, and what certain notifications mean.

Stream Processing

Stream processing is the process of being able to almost instantaneously analyze data that is streaming from one device to another. This method of continuous computation happens as data flows through the system with no compulsory time limitations on the output. With the almost instant flow, systems do not require large amounts of data to be stored.

Stream processing is highly beneficial if the events you wish to track are happening frequently and close together in time. It is also best to utilize if the event needs to be detected right away and responded to quickly. Stream processing, then, is useful for tasks like fraud detection and cybersecurity. If transaction data is stream-processed, fraudulent transactions can be identified and stopped before they are even complete.

Challenges

One of the biggest challenges that organizations face with stream processing is that the system’s long-term data output rate must be just as fast or faster than the long-term data input rate otherwise the system will begin to have issues with storage and memory.

Another challenge is trying to figure out the best way to cope with the huge amount of data that is being generated and moved. In order to keep the flow of data through the system operating at the highest optimal level, it is necessary for organizations to create a plan for how to reduce the number of copies, how to target compute kernels, and how to utilize the cache hierarchy in the best way possible.

Conclusion

While all of these systems have advantages, at the end of the day organizations should consider the potential benefits of each to decide which method is best suited for the use-case.

Although people use the word in different ways, Hadoop refers to an ecosystem of projects, most of which are not processing systems at all. It contains MapReduce, which is a very batch-oriented data processing paradigm.

Spark is also part of the Hadoop ecosystem, I’d say, although it can be used separately from things we would call Hadoop. Spark is a batch processing system at heart too. Spark *Streaming* is a stream processing system.

To me a stream processing system:

* Computes a function of one data element, or a smallish window of recent data
* Computes something relatively simple
* Needs to complete each computation in near-real-time — probably seconds at most
* Computations are generally independent
* Asynchronous – source of data doesn’t interact with the stream processing directly, like by waiting for an answer

A batch processing system to me is just the general case, rather than a special type of processing, but I suppose you could say that a batch processing system:

* Has access to all data
* Might compute something big and complex
* Is generally more concerned with throughput than latency of individual components of the computation
* Has latency measured in minutes or more

I sometimes hear *streaming* used as a sort of synonym for *real-time* . Real-time stuff usually takes the form of needing to respond to an event in milliseconds, as in a synchronous API. This isn’t *streaming* to me.

Q 11. Is fulume running continuesly

<https://data-flair.training/blogs/flume-sink/>

Q 12.what type of configurations do you have for flume sink

The 3 different built in channel types available in Flume are-

MEMORY Channel – Events are read from the source into memory and passed to the sink.

JDBC Channel – JDBC Channel stores the events in an embedded Derby database.

FILE Channel –File Channel writes the contents to a file on the file system after reading the event from a source. The file is deleted only  after the contents are successfully delivered to the sink.

MEMORY Channel is the fastest channel among the three however has the risk of data loss. The channel that you choose completely depends on the nature of the big data application and the value of each event.

13. if incoming data is much more than data size configured in channel what will you do

14.suppose we want to stream 2TB of data what will be the configurations of the flume

To make partition use kafka cluster for this

20.how much resources you given to yarn