



Cloudera Administrator Training for Apache Hadoop





Introduction

Chapter 1



Course Chapters

■ Introduction

- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Chapter Topics

Introduction

- **About This Course**
- About Cloudera
- Course Logistics
- Introductions

Course Objectives (1)

During this course, you will learn:

- **The core technologies of Hadoop**
- **How Cloudera Manager can simplify Hadoop administration**
- **How to deploy a Hadoop cluster**
- **How to run YARN applications, including MapReduce and Spark**
- **How to populate HDFS from external sources using Sqoop and Flume**
- **How to plan your Hadoop cluster hardware and software**
- **What issues to consider when installing Hive and Impala**
- **What issues to consider when deploying Hadoop clients**
- **How to configure HDFS for high availability**

Course Objectives (2)

- **What issues to consider when implementing Hadoop security**
- **How to configure resource scheduling on the cluster**
- **How to maintain your cluster**
- **How to monitor, troubleshoot, and optimize the cluster**

Chapter Topics

Introduction

- About This Course
- **About Cloudera**
- Course Logistics
- Introductions

About Cloudera (1)



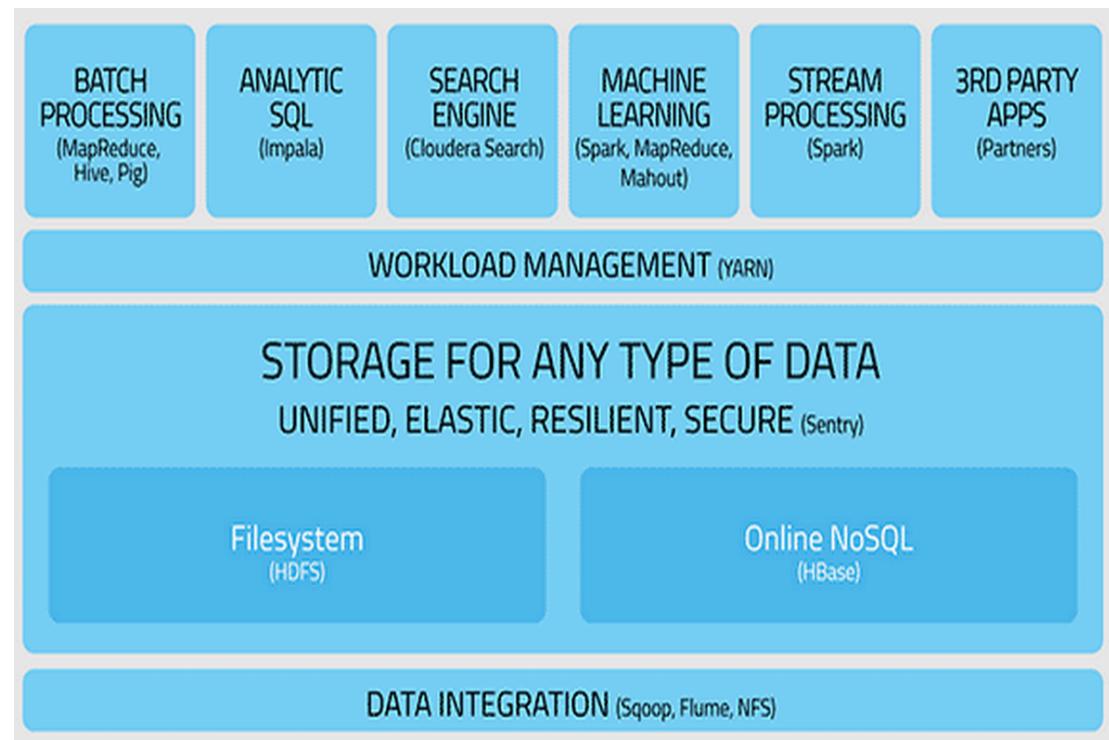
- The leader in Apache Hadoop-based software and services
- Founded by leading experts on Hadoop from Facebook, Yahoo, Google, and Oracle
- Provides support, consulting, training, and certification for Hadoop users
- Staff includes committers to virtually all Hadoop projects
- Many authors of industry standard books on Apache Hadoop projects
 - Tom White, Lars George, Sandy Ryza, etc.

About Cloudera (2)

- **Customers include many key users of Hadoop**
 - Allstate, AMD, Box, BT, CBS Interactive, eBay, Experian, Groupon, Merkle, National Cancer Institute, NetApp, Nokia, Orbitz, Qualcomm, Samsung, Social Security Administration, Trend Micro, Trulia, US Army, ...
- **Cloudera public training:**
 - Cloudera Developer Training for Spark and Hadoop I
 - Cloudera Developer Training for Spark and Hadoop II: Advanced Techniques
 - Data Science at Scale with Spark and Hadoop
 - Cloudera Administrator Training for Apache Hadoop
 - Cloudera Search Training
 - Cloudera Data Analyst Training: Using Pig, Hive, and Impala with Hadoop
 - Cloudera Training for Apache HBase
- **Onsite and custom training is also available**

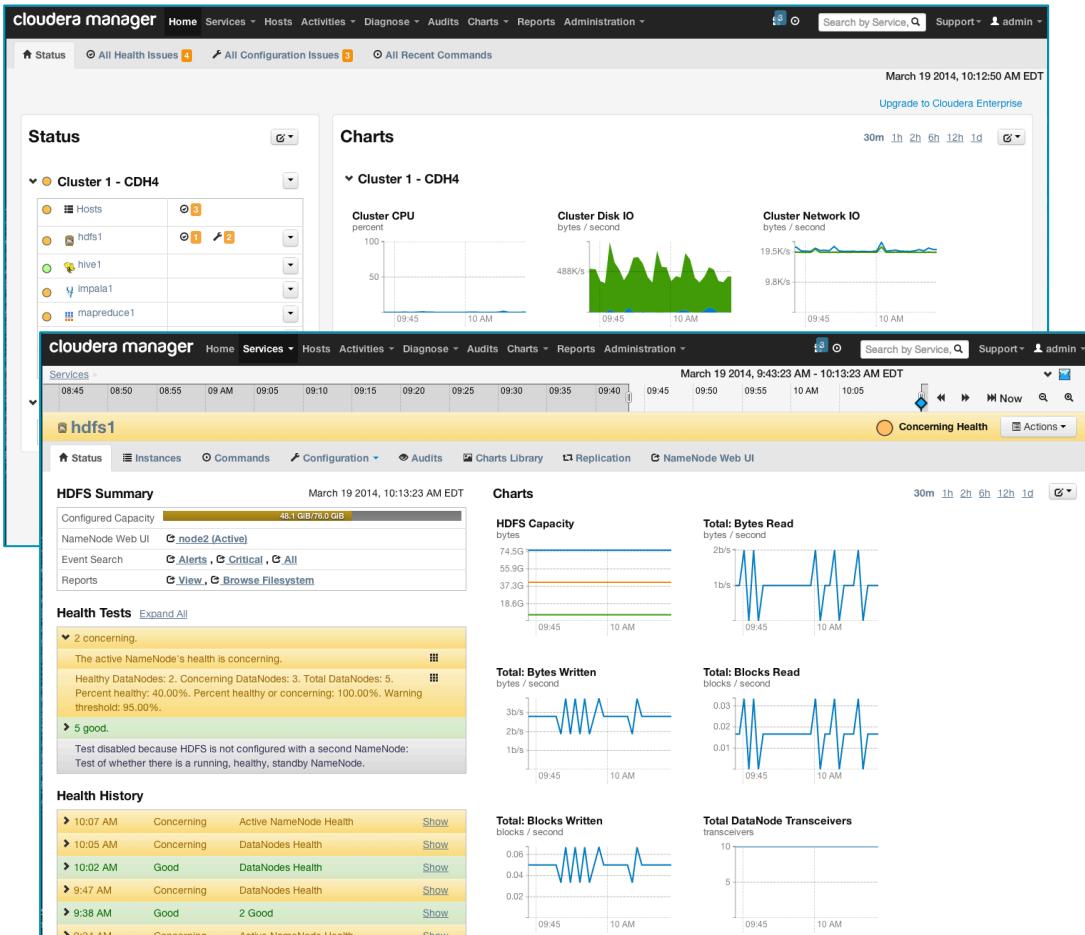
CDH: Cloudera's Distribution, including Apache Hadoop

- **100% open source, enterprise-ready distribution of Hadoop and related projects**
- **The most complete, tested, and widely-deployed distribution of Hadoop**
- **Integrates all the key Hadoop ecosystem projects**
- **Available as RPMs and Ubuntu/Debian/SuSE packages or as a tarball**



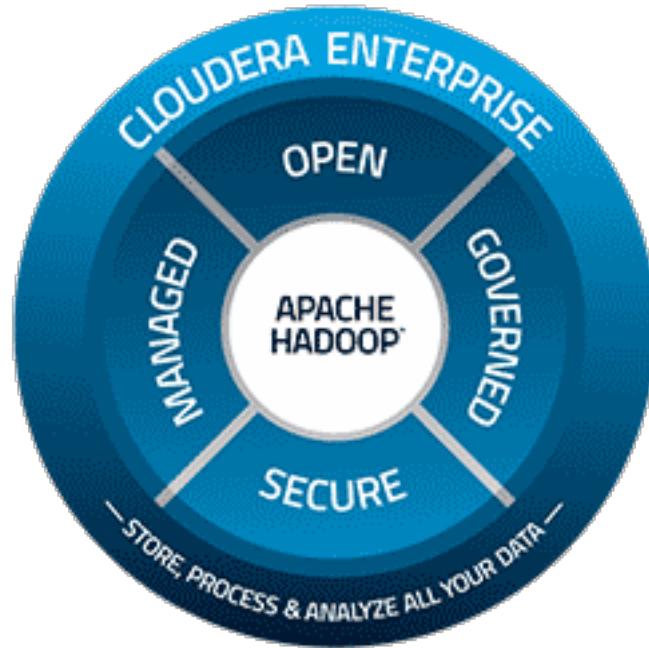
Cloudera Express

- Cloudera Express
 - Free download
- The best way to get started with Hadoop
- Includes CDH
- Includes Cloudera Manager
 - End-to-end administration for Hadoop
 - Deploy, manage, and monitor your cluster



Cloudera Enterprise

- **Cloudera Enterprise**
 - Subscription product including CDH and Cloudera Manager
- **Includes support**
- **Includes extra Cloudera Manager features**
 - Configuration history and rollbacks
 - Rolling updates
 - LDAP integration
 - SNMP support
 - Automated disaster recovery
 - ...and more



Chapter Topics

Introduction

- About This Course
- About Cloudera
- **Course Logistics**
- Introductions

Logistics

- Class start and finish times
- Lunch
- Breaks
- Restrooms
- Wi-Fi access
- Virtual machines
- Can I come in early/stay late?

Your instructor will give you details on how to access the course materials and exercise instructions for the class

Chapter Topics

Introduction

- About This Course
- About Cloudera
- Course Logistics
- **Introductions**

Introductions

- **About your instructor**
- **About you**
 - How much Hadoop experience do you have?
 - Experience as a System Administrator?
 - What platform(s) do you use?
 - What do you expect to gain from this course?



The Case for Apache Hadoop

Chapter 2



Course Chapters

- Introduction
- **The Case for Apache Hadoop**
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

The Case for Apache Hadoop

In this chapter, you will learn:

- **Why Hadoop is needed**
- **What problems Hadoop solves**
- **What comprises Hadoop and the Hadoop ecosystem**

Chapter Topics

The Case for Apache Hadoop

- **Why Hadoop?**
- Fundamental Concepts
- Core Hadoop Components
- Conclusion
- Hands-On Exercise: Network Setup

The Data Deluge (1)

- **We are generating more data than ever**

- Financial transactions
- Sensor networks
- Server logs
- Analytics
- e-mail and text messages
- Social media

The Data Deluge (2)

- **And we are generating data faster than ever**
 - Automation
 - Ubiquitous internet connectivity
 - User-generated content
- **For example, every day**
 - Twitter processes 500 million messages
 - Amazon S3 storage adds more than one billion objects
 - Facebook users generate 4.5 billion comments and “Likes”

Data is Value

- **This data has many valuable applications**
 - Marketing analysis
 - Product recommendations
 - Demand forecasting
 - Fraud detection
 - And many, many more...
- **We must process it to extract that value**

Data Processing Scalability

- **How can we process all that information?**
- **There are actually two problems**
 - Large-scale data storage
 - Large-scale data analysis

Disk Capacity and Price

- We are generating more data than ever before
- Fortunately, the size and cost of storage has kept pace
 - Capacity has increased while price has decreased

Year	Capacity (GB)	Cost per GB (USD)
1997	2.1	\$157
2004	200	\$1.05
2015	3,000	\$0.029

Disk Capacity and Performance

- Disk performance has also increased in the last 15 years
- Unfortunately, transfer rates have not kept pace with capacity

Year	Capacity (GB)	Transfer Rate (MB/s)	Disk Read Time
1997	2.1	16.6	126 seconds
2004	200	56.5	59 minutes
2015	3,000	210	3 hours, 58 minutes

Data Access is the Bottleneck

- **Although we can process data more quickly, *accessing* it is slow**
 - This is true for both reads and writes
- **For example, reading a single 3TB disk takes almost four hours**
 - We cannot process the data until we have read it
 - We are limited by the speed of a single disk
- **We will see Hadoop's solution in a few moments**
 - But first we will examine how we *process* large amounts of data

Monolithic Computing

- **Traditionally, computation has been processor-bound**
 - Intense processing on small amounts of data
- **For decades, the goal was a bigger, more powerful machine**
 - Faster processor, more RAM
- **This approach has limitations**
 - High cost
 - Limited scalability



The Case for Distributed Systems

“In pioneer days they used oxen for heavy pulling, and when one ox couldn’t budge a log, we didn’t try to grow a larger ox. We shouldn’t be trying for bigger computers, but for *more systems* of computers.”

– Grace Hopper



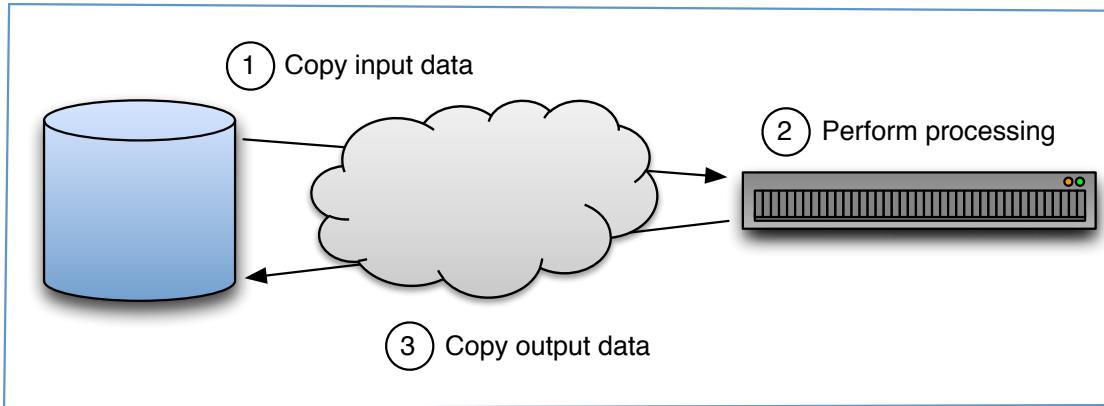
Distributed Computing

- **Modern large-scale processing is distributed across machines**
 - Often hundreds or thousands of nodes
 - Common frameworks include MPI, PVM and Condor
- **Focuses on distributing the processing workload**
 - Powerful compute nodes
 - Separate systems for data storage
 - Fast network connections to connect them

Distributed Computing Processing Pattern

- **Typical processing pattern**

- Step 1: Copy input data from storage to compute node
 - Step 2: Perform necessary processing
 - Step 3: Copy output data back to storage



- **This works fine with relatively small amounts of data**
 - That is, where step 2 dominates overall runtime

Data Processing Bottleneck

- **That pattern does not scale with large amounts of data**
 - More time spent copying data than actually processing it
 - Getting data to the processors is the bottleneck
- **Grows worse as more compute nodes are added**
 - They are competing for the same bandwidth
 - Compute nodes become starved for data

Complexity of Distributed Computing

- **Distributed systems pay for scalability by adding complexity**
- **Much of this complexity involves**
 - Availability
 - Data consistency
 - Event synchronization
 - Bandwidth limitations
 - Partial failure
 - Cascading failures
- **These are often more difficult than the original problem**
 - Error handling often accounts for the majority of the code

System Requirements: Failure Handling

- **Failure is inevitable**
 - We should strive to handle it well
- **An ideal solution should have (at least) these properties**

Failure-Handling Properties of an Ideal Distributed System

Automatic	Job can still complete without manual intervention
Transparent	Tasks assigned to a failed component are picked up by others
Graceful	Failure results only in a proportional loss of load capacity
Recoverable	That capacity is reclaimed when the component is later replaced
Consistent	Failure does not produce corruption or invalid results

More System Requirements

- **Linear horizontal scalability**
 - Adding new nodes should add proportional load capacity
 - Avoid contention by using a “shared nothing” architecture
 - Must be able to expand cluster at a reasonable cost
- **Jobs run in relative isolation**
 - Results must be independent of other jobs running concurrently
 - Although performance can be affected by other jobs
- **Simple programming model**
 - Should support a widely-used language
 - The API must be relatively easy to learn
- **Hadoop addresses these requirements**

Chapter Topics

The Case for Apache Hadoop

- Why Hadoop?
- **Fundamental Concepts**
- Core Hadoop Components
- Conclusion
- Hands-On Exercise: Network Setup

Hadoop: A Radical Solution

- **Traditional distributed computing frequently involves**
 - Complex programming requiring explicit synchronization
 - Expensive, specialized fault-tolerant hardware
 - High-performance storage systems with built-in redundancy
- **Hadoop takes a radically different approach**
 - Inspired by Google's GFS and MapReduce architecture
 - This new approach addresses the problems described earlier

Hadoop Scalability

- **Hadoop aims for linear horizontal scalability**
 - Cross-communication among nodes is minimal
 - Just add nodes to increase cluster capacity and performance
- **Clusters are built from industry-standard hardware**
 - Widely-available and relatively inexpensive servers
 - You can “scale out” later when the need arises

Solution: Data Access Bottleneck

- **Recap: separate storage and compute systems create bottleneck**
 - Can spend more time copying data than processing it
- **Solution: store and process data on the same machines**
 - This is why adding nodes increases capacity and performance
- **Optimization: Use intelligent job scheduling (data locality)**
 - Hadoop tries to process data on the same machine that stores it
 - This improves performance and conserves bandwidth
 - “Bring the computation to the data”

Solution: Disk Performance Bottleneck

- **Recap: a single disk has great capacity but poor performance**
- **Solution: use multiple disks in parallel**
 - The transfer rate of one disk might be 210 *megabytes*/second
 - Almost four hours to read 3 TB of data
 - 1000 such disks in parallel can transfer 210 *gigabytes*/second
 - Less than 15 seconds to read 3TB of data
- **Colocated storage and processing makes this solution feasible**
 - 100-node cluster with 10 disks per node = 1000 disks

Solution: Complex Processing Code

- **Recap: Distributed programming is very difficult**
 - Often done in C or FORTRAN using complex libraries
- **Solution: Use a popular language and a high-level API**
 - MapReduce code is typically written in Java (like Hadoop itself)
 - It is possible to write MapReduce in nearly any language
- **The MapReduce programming model simplifies processing**
 - Deal with one record (key-value pair) at a time
 - Complex details are abstracted away
 - No file I/O
 - No networking code
 - No synchronization

Solution: Fault Tolerance

- **Recap: Distributed systems often use expensive components**
 - In order to minimize the *possibility* of failure
- **Solution: Realize that failure is inevitable**
 - And instead try to minimize the *effect* of failure
 - Hadoop satisfies all the requirements we discussed earlier
- **Machine failure is a regular occurrence**
 - A server might have a mean time between failures (MTBF) of 5 years (~1825 days)
 - Equates about one failure per day in a 2,000 node cluster

Chapter Topics

The Case for Apache Hadoop

- Why Hadoop?
- Fundamental Concepts
- **Core Hadoop Components**
- Conclusion
- Hands-On Exercise: Network Setup

Core Hadoop Components

- **Hadoop is a system for large-scale data processing**
- **Hadoop provides**
 - HDFS for data storage
 - The extensible YARN framework
 - For application scheduling and resource management
 - Includes MapReduce version 2 for data processing
- **Plus the infrastructure needed to make them work, including**
 - Filesystem utilities
 - Application scheduling and monitoring
 - Web UI

The Hadoop Ecosystem

- **Many related tools integrate with Hadoop**
 - Data processing: Spark
 - Data analysis: Hive, Pig, and Impala
 - Data discovery: Solr (Cloudera Search)
 - Machine learning: MLlib, Mahout, and others
 - Data ingestion: Sqoop, Flume, Kafka
 - Coordination: ZooKeeper
 - User experience: Hue
 - Workflow management: Oozie
 - Cluster management: Cloudera Manager
- **These are not considered “core Hadoop”**
 - Rather, they are part of the “Hadoop ecosystem”
 - Many are also open source Apache projects
 - We will learn about several of these later in the course

Chapter Topics

The Case for Apache Hadoop

- Why Hadoop?
- Fundamental Concepts
- Core Hadoop Components
- **Conclusion**
- Hands-On Exercise: Network Setup

Essential Points

- We are generating more data – and faster – than ever before
- We can store and process the data, but there are problems using existing techniques
 - Accessing the data from disk is a bottleneck
 - In distributed systems, getting data to the processors is a bottleneck
- Hadoop eliminates the bottlenecks by storing and processing data on the same machine

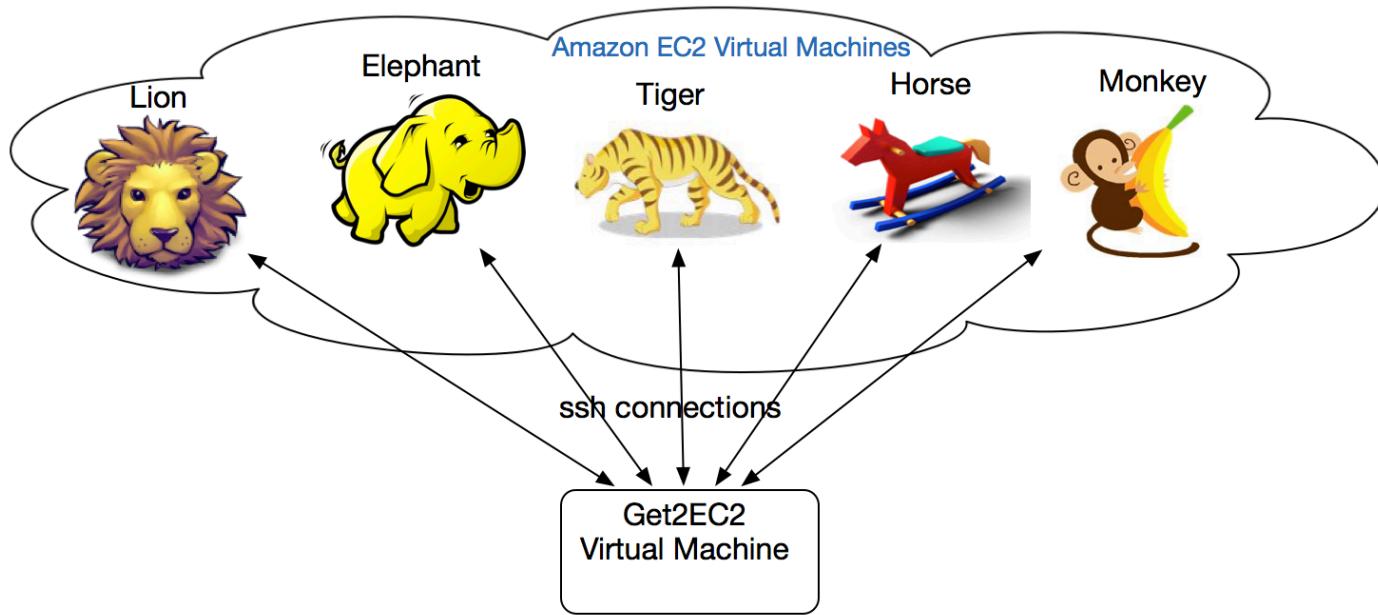
Chapter Topics

The Case for Apache Hadoop

- Why Hadoop?
- Fundamental Concepts
- Core Hadoop Components
- Conclusion
- **Hands-On Exercise: Network Setup**

Hands-On Exercise: Configuring Networking

- In this Hands-On Exercise, you will configure network connectivity for your environment: *local* or *cloud*
- Please refer to the Hands-On Exercise Manual



- Note: Each machine in the exercises is assigned an animal name for convenience



Hadoop Cluster Installation

Chapter 3



Course Chapters

- Introduction
- The Case for Apache Hadoop
- **Hadoop Cluster Installation**
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Hadoop Installation and Initial Configuration

In this chapter, you will learn:

- **The rationale for, and benefits of, a cluster management solution**
- **Cloudera Manager features, options, and requirements**
- **How to install Cloudera Manager**
- **CDH cluster installation options**
- **How to install Hadoop**

Chapter Topics

Hadoop Cluster Installation

- **Rationale for a Cluster Management Solution**
- Cloudera Manager Features
- Cloudera Manager Installation
- Hands-On Exercise: Installing Cloudera Manager Server
- CDH Installation
- Conclusion
- Hands-On Exercise: Creating a Hadoop Cluster

Hadoop Cluster Overview

- Hadoop daemons run on a cluster of machines
- The Hadoop Distributed File System (HDFS) is used to distribute data amongst the nodes
- Computational frameworks such as MapReduce, Spark, and Impala bring the computing to the data
- To realize the benefits of Hadoop you must deploy Hadoop daemons across a sizable cluster of machines
 - Many organizations maintain multiple clusters, each with hundreds or thousands of nodes

Rationale for a Hadoop Cluster Management Application

- **Apache Hadoop is a large, complex system**
 - Installing, configuring, monitoring, managing, upgrading, and troubleshooting a distributed Apache Hadoop cluster (or multiple clusters) is non-trivial
- **A manual management approach is error-prone and does not scale or offer the benefits of a fully developed management application**
- **A management tool is essential for any cluster of reasonable size**
- **Cloudera Manager is the preeminent Hadoop management tool**
 - Free to download
 - Free to use
 - No commitment to Cloudera

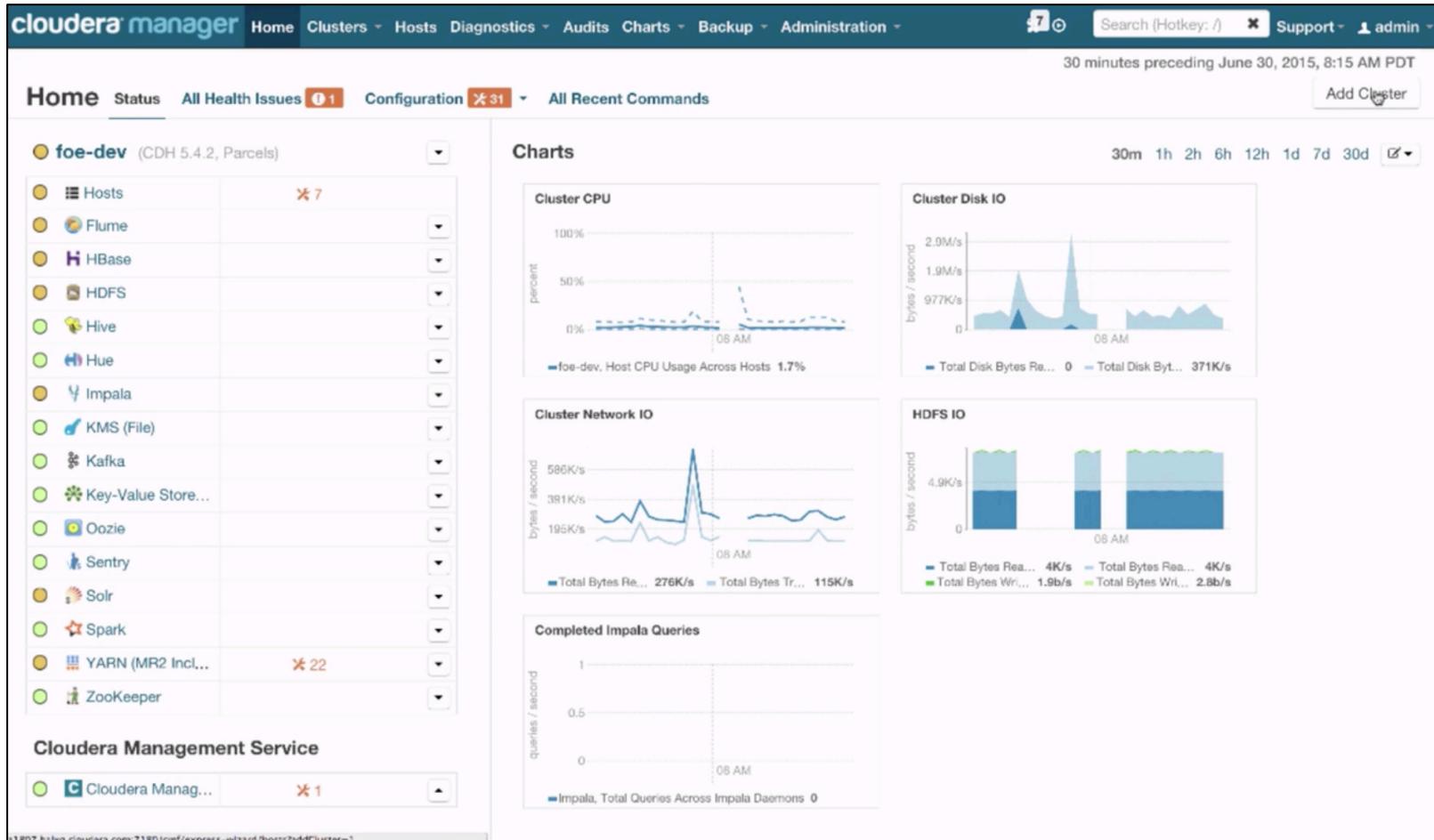
Chapter Topics

Hadoop Cluster Installation

- Rationale for a Cluster Management Solution
- **Cloudera Manager Features**
- Cloudera Manager Installation
- Hands-On Exercise: Installing Cloudera Manager Server
- CDH Installation
- Conclusion
- Hands-On Exercise: Creating a Hadoop Cluster

What Is Cloudera Manager?

- An application designed to meet Hadoop enterprise-level users' needs



Cloudera Manager Features

- **Automated deployment**
 - Automatically install and configure Hadoop services on hosts
 - Cloudera Manager sets recommended default parameters
 - Easy to stop and start services on master and worker nodes
- **Manage a wide range of Hadoop and Hadoop ‘ecosystem’ services**
 - Including HDFS, YARN, MapReduce, Spark, Hive, Pig, Impala, Flume, Oozie, Soop, ZooKeeper, Hue, HBase, Cloudera Search, and more
- **Diagnose and resolve issues more quickly**
 - Daemon logging is aggregated and searchable across the cluster
- **Manage user and group access to the cluster(s)**
- **Monitor cluster health and performance, track events**
 - Real-time monitoring, charts, custom reporting, email alerts

Cloudera Manager – Two Editions

■ Cloudera Express

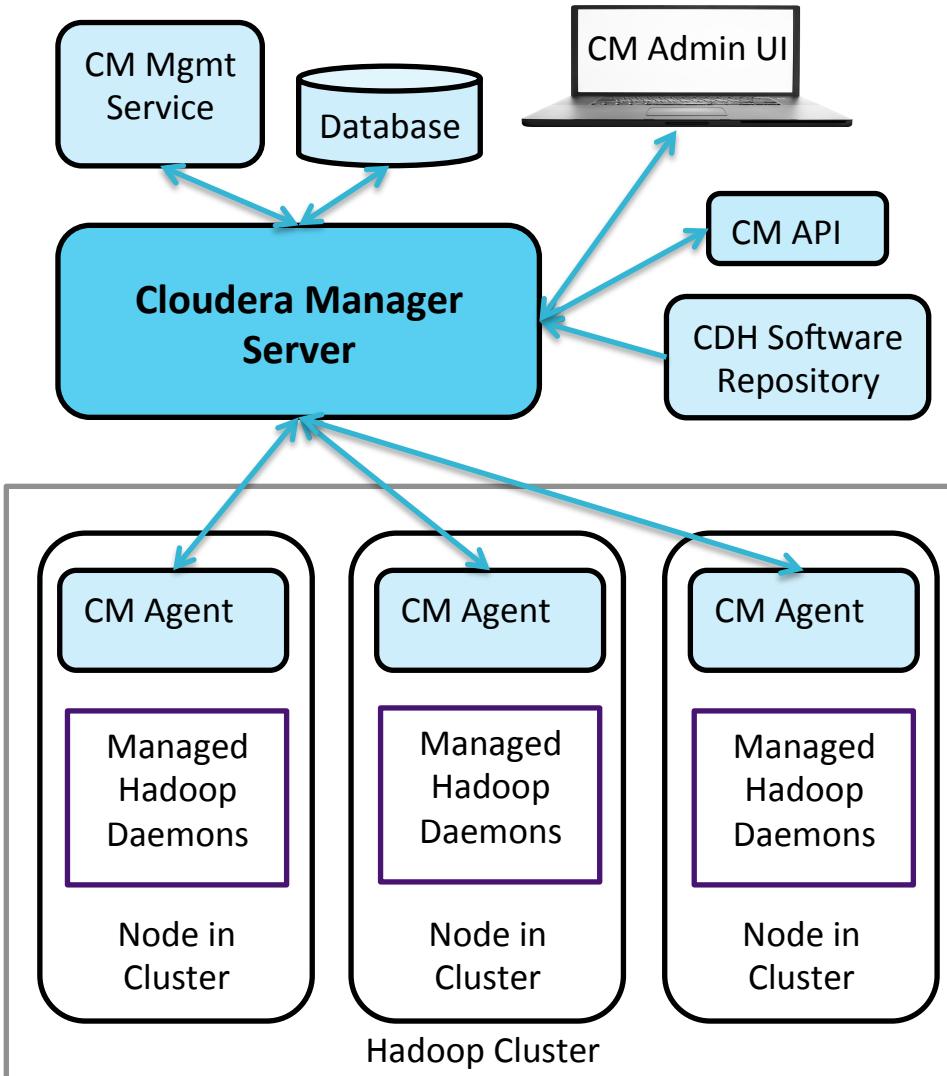
- Free download
- Manage a cluster of any size
- Easy upgrade to Cloudera Enterprise
 - A 60-day Enterprise trial is free

■ Cloudera Enterprise

- Includes support
- Includes extra features and functionality for Enterprise Data Hubs
 - Rolling upgrades
 - SNMP support
 - LDAP integration
 - Configuration history and rollbacks
 - Operational reports
 - Automated disaster recovery

	Cloudera Express	Cloudera Enterprise
Subscription	Free	Annual
Deployment & Configuration	•	•
Management	•	•
Monitoring	•	•
Diagnostic Tools	•	•
Advanced Mgt Features		•
Support		•

Basic Topology of Cloudera Manager



▪ Cloudera Manager Server

- Installed outside the cluster
- Runs service monitor and activity monitor daemons
- Stores cluster configuration information in a database
- Sends configuration information and commands to the agents over HTTP(S)

▪ Cloudera Manager Agents

- Receive updated configurations from server
- Start and stop Hadoop daemons, collect statistics
- Heartbeat status to the server

Chapter Topics

Hadoop Cluster Installation

- Rationale for a Cluster Management Solution
- Cloudera Manager Features
- **Cloudera Manager Installation**
- Hands-On Exercise: Installing Cloudera Manager Server
- CDH Installation
- Conclusion
- Hands-On Exercise: Creating a Hadoop Cluster

Cloudera Manager Deployment – An Overview

- 1. Install and configure the database, install the Oracle JDK**
 - Database should be external for Production deployments
 - Embedded PostgreSQL is OK for ‘proof of concept’ work
- 2. Ensure access to the Cloudera software repositories**
 - For Cloudera Manager
 - For CDH
- 3. Install Cloudera Manager and agents**
- 4. Install the CDH Parcel services or RPMs for the functionality required on each host in the cluster**
 - HDFS
 - YARN (Including MR2)
 - ...

Deploying on Multiple Machines

- When commissioning multiple machines, use an automated operating system deployment tool
 - Red Hat's Kickstart
 - Debian Fully Automatic Installation
 - Dell Crowbar
 - ...
- You might optionally use a tool to manage the underlying operating system
 - Puppet
 - Chef
 - ...
- Use Cloudera Manager to install Hadoop and manage the Hadoop cluster

Cloudera Manager 5 Requirements (1)

- **Supported Operating Systems (all 64-bit)**

- Red Hat Enterprise Linux/Centos 5.7, 6.4, 6.5, 6.6
- Oracle Enterprise Linux 5.6, 6.4, 6.5, 6.6
- SUSE Linux Enterprise Server 11 Service Pack 2 or later
- Debian 7.0, 7.1
- Ubuntu 12.04, 14.04

- **Supported Browsers**

- Internet Explorer 9 or later
- Google Chrome
- Safari 5 or later
- Firefox 24 or 31

Cloudera Manager 5 Requirements (2)

- **Supported JDKs**

- Oracle JDK 1.7.0_55, 1.7.0_67 or higher, 1.8.0_40 or higher

- **Supported databases**

- MySQL 5.5 and 5.6
 - Oracle 11g Release 2
 - PostgreSQL 8.4, 9.2, and 9.3

- **Supported Cloudera Distribution of Hadoop (CDH)**

- CDH 5.0 or later
 - CDH 4.0 or later (now end-of-life, and therefore deprecated)

Installing Cloudera Manager

- **Cloudera Manager installation options**
 - Package distribution (recommended)
 - Package repository available at
<http://archive.cloudera.com/cm5/>
 - Install using yum, zypper, or apt-get
 - Binary distribution
 - Download binary from
<http://www.cloudera.com/downloads>
 - Run the installer from the command line
- **After installation, complete subsequent configuration through the Web UI**
 - Access via `http://<manager_host>:7180`
- **Documentation**
 - For detailed information about installing Cloudera Manager, refer to
<http://docs.cloudera.com/>

Cloudera Manager Deployment Options

- **Conventional mode**
 - Cloudera Manager Agent runs as the root OS user
- **Single user mode**
 - Available beginning with Cloudera Manager 5.3 and CDH 5.2
 - Cloudera Manager Agent, and all processes run by services managed by Cloudera Manager, are started as a single configured user and group
 - Default user: cloudera-scm
 - Prioritizes isolation between Hadoop and the rest of the system
 - De-prioritizes isolation between Hadoop process running on the system
 - Limitations and manual configurations apply – see documentation for details
- **The choice made applies to all clusters managed by the Cloudera Manager deployment**

Cloudera Manager Log Locations

- **On the machine where Cloudera Manager Server is installed**
 - /var/log/cloudera-scm-server/cloudera-scm-server.log
- **On all machines in the cluster**
 - /var/log/cloudera-scm-agent/cloudera-scm-agent.log

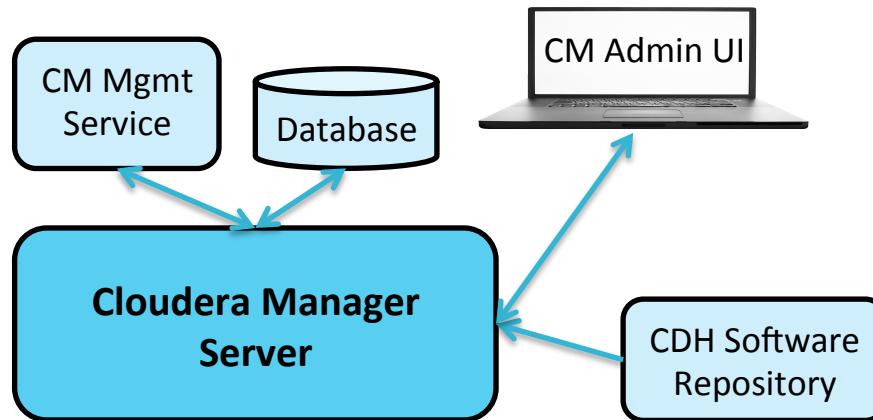
Chapter Topics

Hadoop Cluster Installation

- Rationale for a Cluster Management Solution
- Cloudera Manager Features
- Cloudera Manager Installation
- **Hands-On Exercise: Installing Cloudera Manager Server**
- CDH Installation
- Conclusion
- Hands-On Exercise: Creating a Hadoop Cluster

Hands-On Exercise: Installing Cloudera Manager Server

- In this Hands-On Exercise, you will configure MySQL, install Cloudera Manager Server, start Cloudera Manager Server, and review the logs
- Please refer to the Hands-On Exercise Manual
- Resources installed or available after exercise completion:



Chapter Topics

Hadoop Cluster Installation

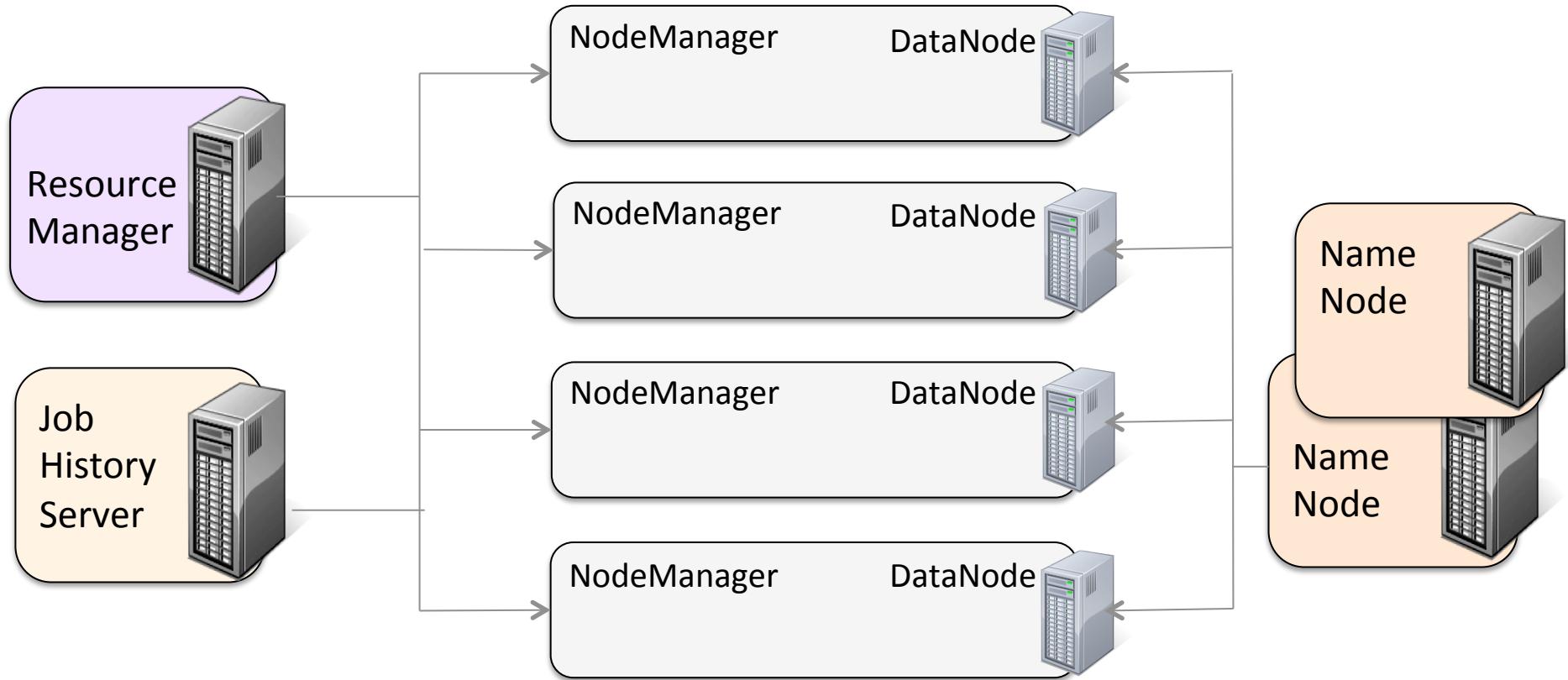
- Rationale for a Cluster Management Solution
- Cloudera Manager Features
- Cloudera Manager Installation
- Hands-On Exercise: Installing Cloudera Manager Server
- **CDH Installation**
- Conclusion
- Hands-On Exercise: Creating a Hadoop Cluster

Cluster Installation Overview

- **Install Cloudera Manager Server**
- **Specify the version of Cloudera Manager Server**
 - Login to the Cloudera Manager Web UI
 - Choose a version of Cloudera Manager
 - Cloudera Express
 - Cloudera Enterprise (Data Hub Edition Trial)
 - Cloudera Enterprise
- **Create a CDH cluster**
 - Using the Cloudera Manager Web UI wizard
 - Specify hosts to include in the cluster
 - Select a CDH repository
 - Install the Cloudera Manager agent to each host
 - Install select CDH services to select hosts

Basic Hadoop Cluster Installation: HDFS and YARN (MR2 Included)

- **YARN – Resource Manager, Job History Server, and many NodeManagers**
- **HDFS – Name Node(s) and many DataNodes**



CDH Installation Options

- **Cloudera Manager deployment**

- Installation Path A
 - Automated installation by Cloudera Manager
 - For Proof of Concept (PoC) installations only
 - Installation Path B
 - Manual installation using Cloudera Manager packages or parcels
 - This is the *recommended* path
 - Installation Path C
 - Manual installation using Cloudera Manager tarballs

- **Unmanaged deployment**

- Entirely manual installation, configuration, operations
 - Difficult to manage with even a small cluster

CDH Software Distribution: Packages or Parcels

- **Cloudera Manager will install the selected Hadoop services on cluster nodes for you**
- **Installation is via packages or parcels**
- **Packages**
 - RPM packages (Red Hat, CentOS), Ubuntu packages, etc.
 - Automatically create needed users/groups, init scripts
- **Parcels (recommended)**
 - Cloudera Manager-specific. All the benefits of packages, plus:
 - Allows easy upgrading with minimal down-time
 - Allows easy rolling upgrades (Cloudera Enterprise)
 - Installation of CDH without sudo – installation handled by the Cloudera Manager Agent

No Internet Access Required for Cluster Nodes

- **Often, cluster nodes do not have Internet access**
 - For security or other reasons
- **Cloudera Manager can install CDH on cluster nodes using a local CDH repository**
 - Mirror the CDH package or Parcel repository on a local machine
 - Configure Cloudera Manager to use the local repository during installation and upgrade

Distribute the Daemons

- **Cloudera Manager CDH installation wizard will suggest specific Hadoop daemons be installed to specific cluster nodes**
 - Based on available resources
 - It is easy to override suggestions
- **Not all daemons run on each machine**
 - NameNode, ResourceManager, JobHistoryServer ('master' daemons)
 - One per cluster, unless running in an HA configuration
 - Secondary NameNode
 - One per cluster in a non-HA environment
 - DataNodes, NodeManagers
 - On each data node in the cluster
 - Exception: for small clusters (less than 10 - 20 nodes), it is acceptable for more than one of the master daemons to run on the same physical node

Starting the Hadoop Daemons

- Always manage Hadoop daemon start/stop/restart from the Admin Console

Service	Instance	Name	Links	Process Status	PID	Uptime	Full log file	Stderr	Stdout
None	None	deploy-client-config		Stopped				Stderr ↗	Stdout ↗
HDFS	DataNode	hdfs-DATANODE	DataNode Web UI ↗	Running	2742	16.0h	Full log file ↗	Stderr ↗	Stdout ↗
HDFS	Failover Controller	hdfs-FAILOVERCONTROLLER		Running	2846	16.0h	Full log file ↗	Stderr ↗	Stdout ↗
HDFS	JournalNode	hdfs-JOURNALNODE	JournalNode Web UI ↗	Running	2827	16.0h	Full log file ↗	Stderr ↗	Stdout ↗
HDFS	NameNode	hdfs-NAMENODE	NameNode Web UI ↗	Running	2785	16.0h	Full log file ↗	Stderr ↗	Stdout ↗
YARN (MR2 Included)	NodeManager	yarn-NODEMANAGER	NodeManager Web UI ↗	Running	2937	16.0h	Full log file ↗	Stderr ↗	Stdout ↗
ZooKeeper	Server	zookeeper-server		Running	2883	16.0h	Full log file ↗	Stderr ↗	Stdout ↗

- In Cloudera Manager managed clusters, the CDH services are not registered at the OS level with init scripts (e.g., `$ chkconfig` will not show them)
 - The daemons run as java processes managed by the CM agents

Chapter Topics

Hadoop Cluster Installation

- Rationale for a Cluster Management Solution
- Cloudera Manager Features
- Cloudera Manager Installation
- Hands-On Exercise: Installing Cloudera Manager Server
- CDH Installation
- **Conclusion**
- Hands-On Exercise: Creating a Hadoop Cluster

Essential Points

- **A cluster management solution is essential for production-size clusters**
 - Cloudera Manager is the preeminent cluster management application
 - The Cloudera Express version is free to download and use
- **There are many CDH installation options available**
 - An external database, and the use of parcels, are recommended
- **Plan to distribute hadoop daemons across the cluster**
 - Deploy each master daemon on its own node

Chapter Topics

Hadoop Cluster Installation

- Rationale for a Cluster Management Solution
- Cloudera Manager Features
- Cloudera Manager Installation
- Hands-On Exercise: Installing Cloudera Manager Server
- CDH Installation
- Conclusion
- **Hands-On Exercise: Creating a Hadoop Cluster**

Hands-On Exercise: Installing a Hadoop Cluster

- In this Hands-On Exercise, you will create a Hadoop cluster with four nodes
- Please refer to the Hands-On Exercise Manual
- Cluster deployment after exercise completion:

					
	elephant	horse	monkey	tiger	lion
HDFS NameNode	✓				
HDFS SecondaryNameNode				✓	
HDFS DataNode	✓	✓	✓	✓	
HDFS Balancer		✓			
YARN (MR2 Included) NodeManager	✓	✓	✓	✓	
YARN (MR2 Included) ResourceManager		✓			
YARN (MR2 Included) JobHistory Server				✓	
Cloudera Manager Server					✓
Cloudera Manager Server Database					✓
Cloudera Management Services					✓
Cloudera Manager Agent	✓	✓	✓	✓	✓



The Hadoop Distributed File System (HDFS)

Chapter 4



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- **The Hadoop Distributed File System (HDFS)**
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

HDFS

In this chapter, you will learn:

- **What features HDFS provides**
- **How HDFS reads and writes files**
- **How the NameNode uses memory**
- **How Hadoop provides file security**
- **How to use the NameNode Web UI**
- **How to use the Hadoop File Shell**

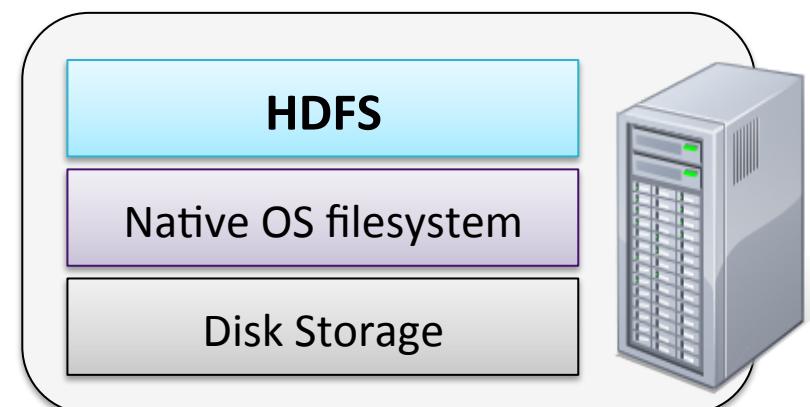
Chapter Topics

HDFS

- **HDFS Features**
- Writing and Reading Files
- NameNode Memory Considerations
- Overview of HDFS Security
- Web UIs for HDFS
- Using the Hadoop File Shell
- Conclusion
- Hands-On Exercise: Working with HDFS

HDFS: The Hadoop Distributed File System

- **HDFS is a filesystem written in Java**
 - Based on Google's GFS (Google File System)
- **Sits on top of a native filesystem**
 - Such as ext3, ext4, or xfs
- **Provides redundant storage for massive amounts of data**
 - Using industry-standard hardware
- **At load time, data is distributed across all nodes**
 - Provides for efficient processing



HDFS Features

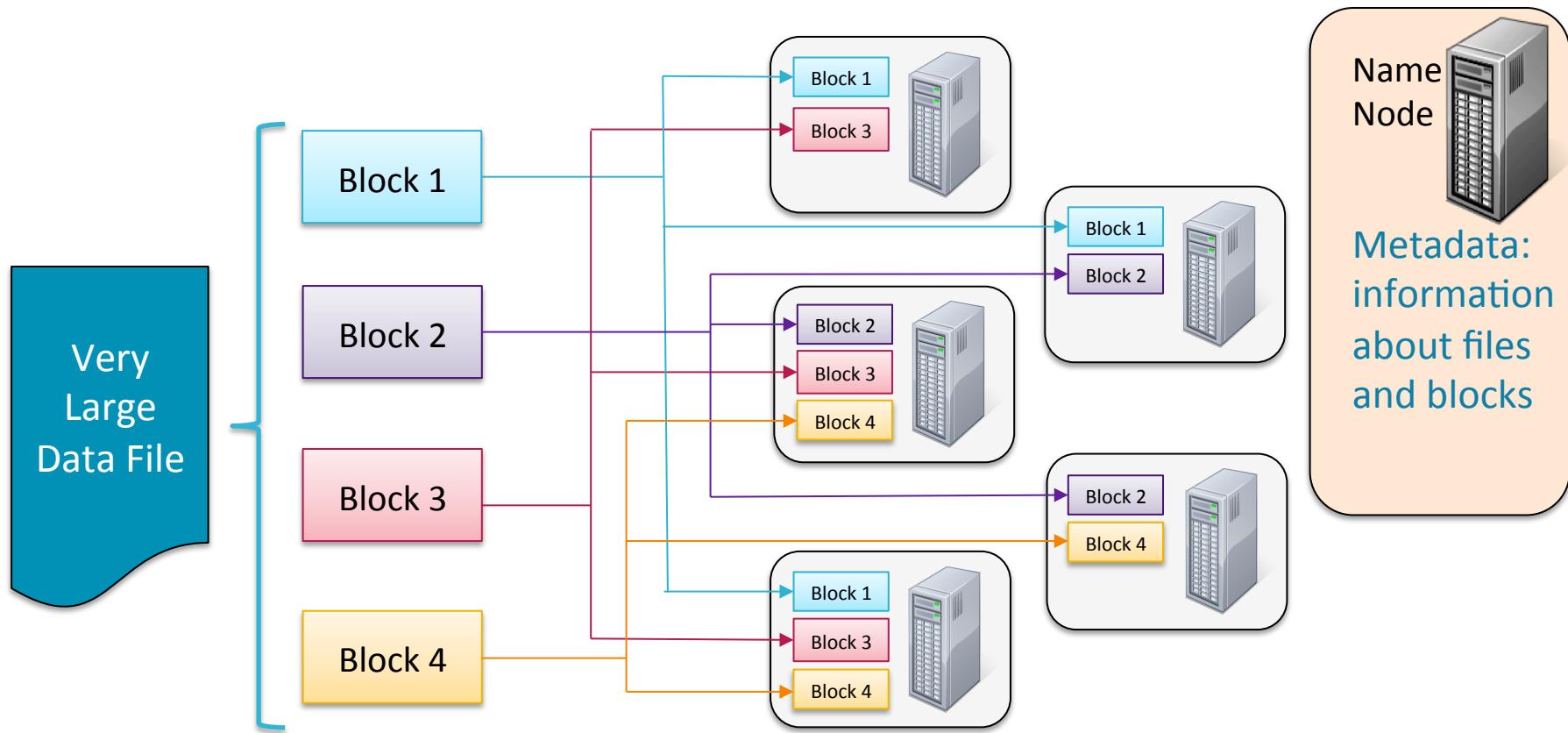
- **High performance**
- **Fault tolerance**
- **Relatively simple centralized management**
 - Master-worker architecture
- **Security**
 - Two levels from which to choose
- **Optimized for distributed processing**
 - Data locality
- **Scalability**

HDFS Design Assumptions

- Components will fail
- “Modest” number of large files
 - Millions of large files, not hundreds of millions of small files
 - Each file is likely to be 100MB or larger
 - Multi-gigabyte files typical
- Files are write-once
 - Data can be appended to a file, but the file’s existing contents cannot be changed
- Large streaming reads
 - Favor high sustained throughput over low latency

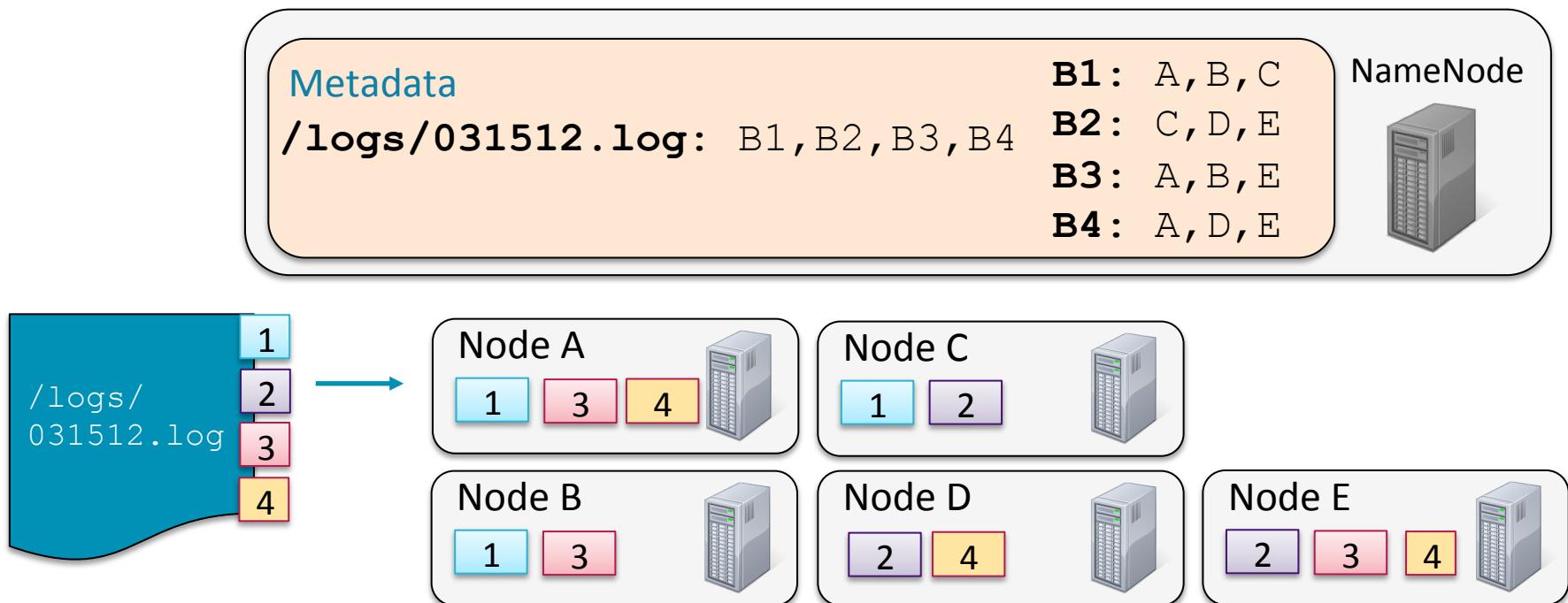
HDFS Blocks

- When a file is added to HDFS, it is split into blocks
 - Similar concept to native filesystems, but *much* larger block size
 - Default block size is 128MB (configurable)



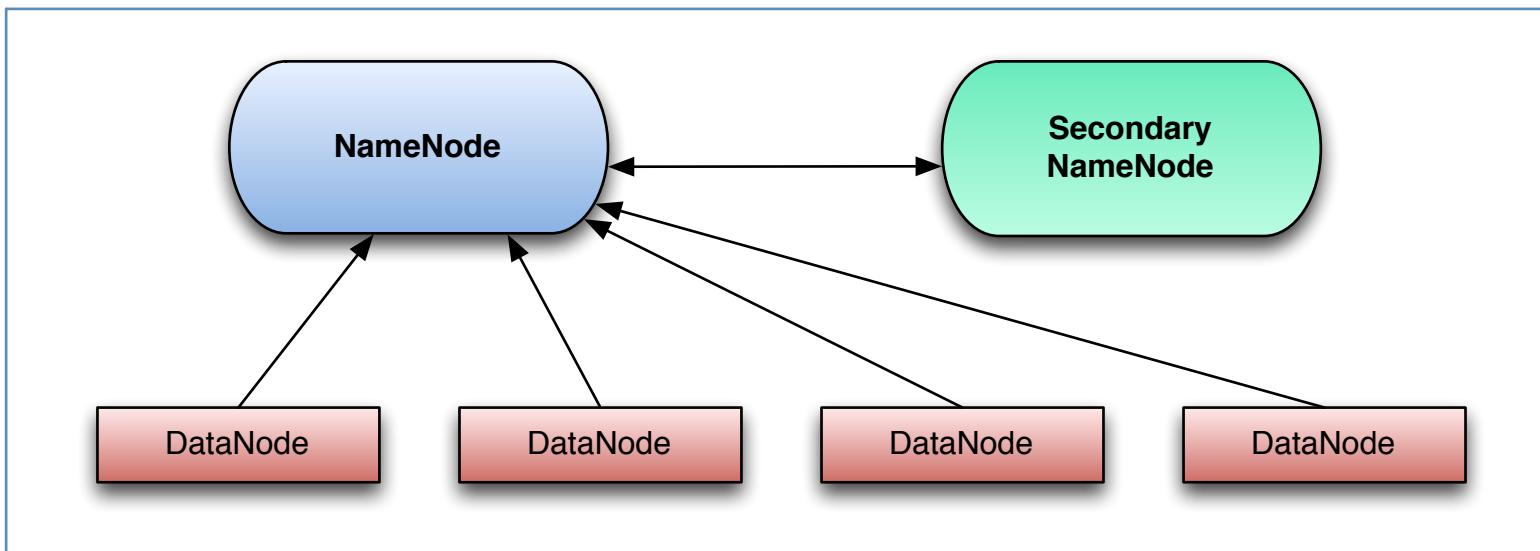
HDFS Replication

- **Blocks are replicated to nodes throughout the cluster**
 - Based on the *replication factor* (default is three)
- **Replication increases reliability and performance**
 - Reliability: data can tolerate loss of all but one replica
 - Performance: more opportunities for data locality



HDFS Without High Availability

- You can deploy HDFS with or without high availability
- Without high availability, there are three daemons
 - NameNode (master)
 - SecondaryNameNode (master)
 - DataNode (worker)



The HDFS NameNode

- **The NameNode holds all *metadata* in RAM**
 - Information about file locations in HDFS
 - Information about file ownership and permissions
 - Names of the individual blocks
 - Locations of the blocks
- **Metadata is stored on disk and read when the NameNode daemon starts up**
 - Filename is `fsimage`
 - Note: block locations are *not* stored in `fsimage`
- **Changes to the metadata are stored in RAM**
 - Changes are also written to an edits log
 - Full details later

The Worker Nodes

- Actual contents of the files are stored as *blocks* on the worker nodes
- Each worker node runs a DataNode daemon
 - Controls access to the blocks
 - Communicates with the NameNode
- Blocks are simply files on the worker nodes' underlying filesystem
 - Named `blk_xxxxxxx`
 - Nothing on the worker node provides information about what underlying file the block is a part of
 - That information is *only* stored in the NameNode's metadata
- Each block is stored on multiple different nodes for redundancy
 - Default is three replicas

The HDFS SecondaryNameNode: Caution!

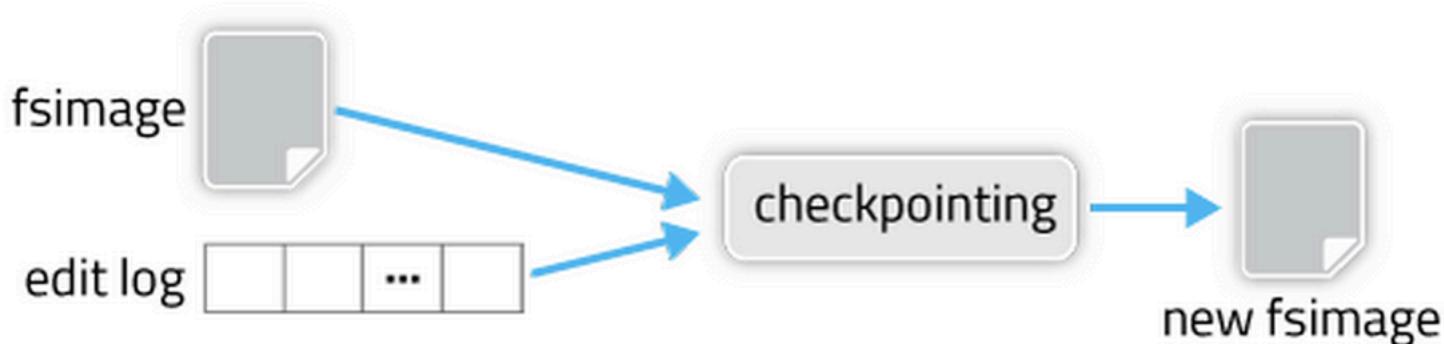
- **The SecondaryNameNode is *not* a failover NameNode!**
 - It performs memory-intensive administrative functions for the NameNode
 - NameNode keeps information about files and blocks (the *metadata*) in memory
 - NameNode writes metadata changes to an `edit log`
 - Secondary NameNode periodically combines a prior snapshot of the file system metadata and edit log into a new snapshot
 - New snapshot is transmitted back to the NameNode
- **SecondaryNameNode should run on a separate machine in a large installation**
 - It requires as much RAM as the NameNode
- **SecondaryNameNode only exists when high availability is not configured**

File System Metadata Snapshot and Edit Log

- **The `fsimage` file contains a file system metadata snapshot**
 - It is *not* updated at every write
- **HDFS write operations are recorded in the NameNode's edit log**
 - The NameNode's in-memory representation of the file system metadata is also updated
- **Applying all changes in the `edits` file(s) during a NameNode restart could take a long time**
 - The files could also grow to be huge

Checkpointing the File System Metadata

- The SecondaryNameNode periodically constructs a checkpoint using this process:
 1. Compacts information in the edits log
 2. Merges it with the most recent `fsimage` file
 3. Clears the edits log
- Benefit: faster NameNode restarts
 - The NameNode can use the latest checkpoint and apply the contents of the smaller edits log

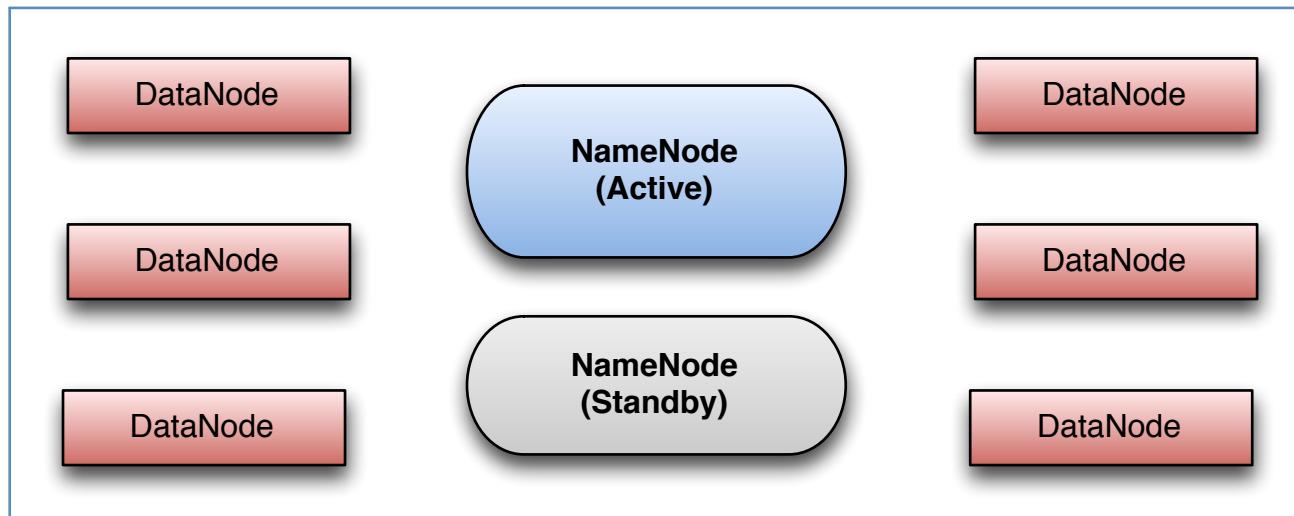


Single Point of Failure

- **In this mode of operation, each Hadoop cluster has a single NameNode**
 - The Secondary NameNode is *not* a failover NameNode
- **The NameNode is a single point of failure (SPOF)**
- **In practice, this is not a major issue**
 - HDFS will be unavailable until NameNode is replaced
 - There is very little risk of data loss for a properly managed system
- **Recovering from a failed NameNode is relatively easy**
 - We will discuss this process in detail later

HDFS With High Availability

- Deploy HDFS with high availability to eliminate the NameNode SPOF
- Two NameNodes: one active and one standby
 - Standby NameNode takes over when active NameNode fails
 - Standby NameNode also does checkpointing (SecondaryNameNode no longer needed)



HDFS Read Caching

- **Applications can instruct HDFS to *cache* blocks of a file**
 - Blocks are stored on the DataNode in off-heap RAM
 - Can result in significant performance gains for subsequent reads
 - Most useful for applications where the same file will be read multiple times
- **It is possible for a user to manually cache files**
 - Impala is caching aware
 - Tables can be cached from within the Impala shell, or set to be cached when they are created
 - Caching-aware applications will attempt to read a block from the node on which it is cached
- **HDFS caching provides benefits over standard OS-level caching**
 - Avoids memory-to-memory copying

Configuring HDFS Read Caching

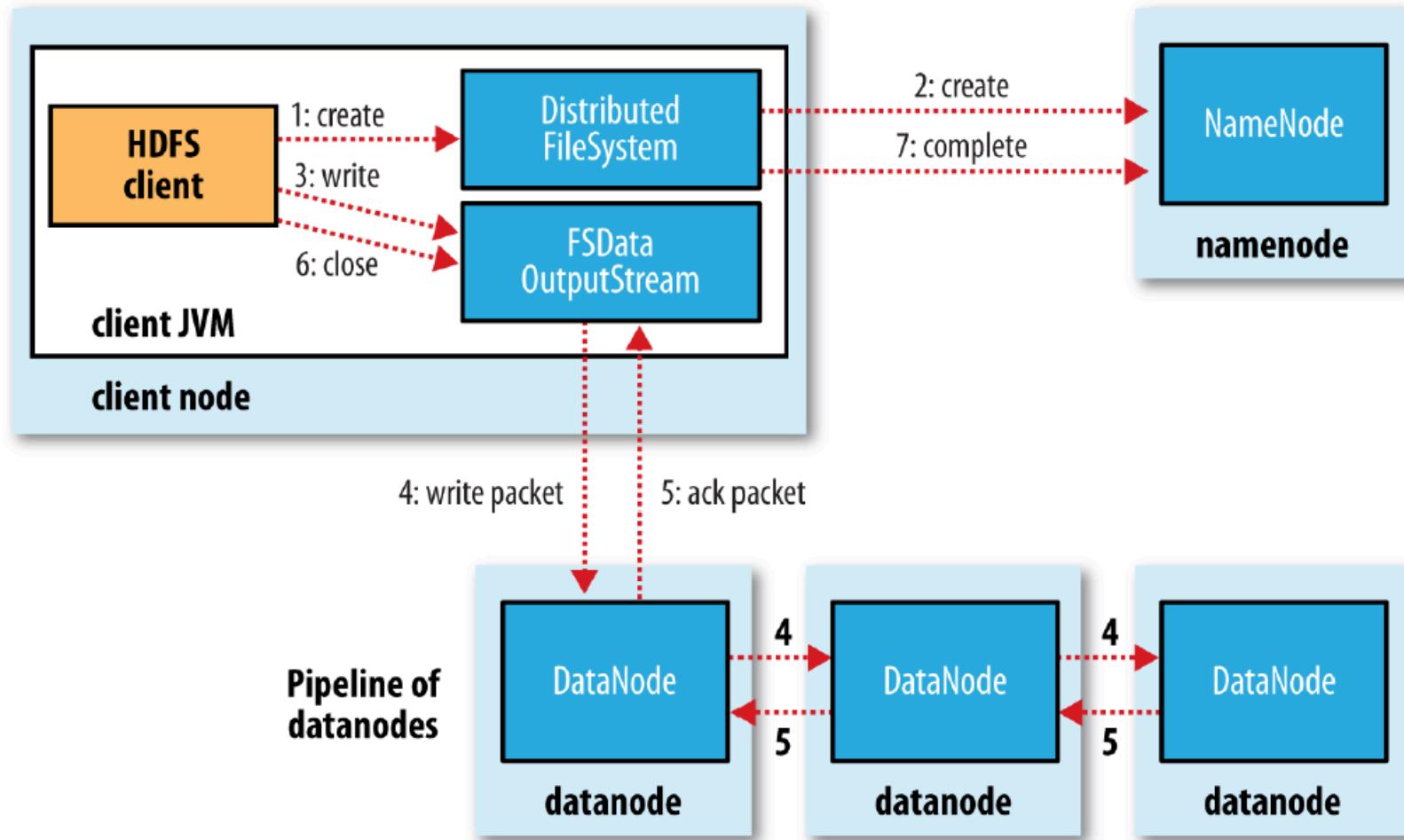
- **Cloudera Manager enables HDFS Read Caching by default**
- **Amount of RAM per DataNode to use for caching is controlled by `dfs.datanode.max.locked.memory`**
 - Can be configured per role group
 - Default is 4GB
 - Set to 0 to disable caching

Chapter Topics

HDFS

- HDFS Features
- **Writing and Reading Files**
- NameNode Memory Considerations
- Overview of HDFS Security
- Web UIs for HDFS
- Using the Hadoop File Shell
- Conclusion
- Hands-On Exercise: Working with HDFS

Anatomy of a File Write (1)



Anatomy of a File Write (2)

- 1. Client connects to the NameNode**
- 2. NameNode places an entry for the file in its metadata, returns the block name and list of DataNodes to the client**
- 3. Client connects to the first DataNode and starts sending data**
- 4. As data is received by the first DataNode, it connects to the second and starts sending data**
- 5. Second DataNode similarly connects to the third**
- 6. ack packets from the pipeline are sent back to the client**
- 7. Client reports to the NameNode when the block is written**

Anatomy of a File Write (3)

- **If a DataNode in the pipeline fails**
 - The pipeline is closed
 - A new pipeline is opened with the two good nodes
 - The data continues to be written to the two good nodes in the pipeline
 - The NameNode will realize that the block is under-replicated, and will re-replicate it to another DataNode
- **As blocks of data are written, the client calculates a checksum for each block**
 - Sent to the DataNode along with the data
 - Written together with each data block
 - Used to ensure the integrity of the data when it is later read

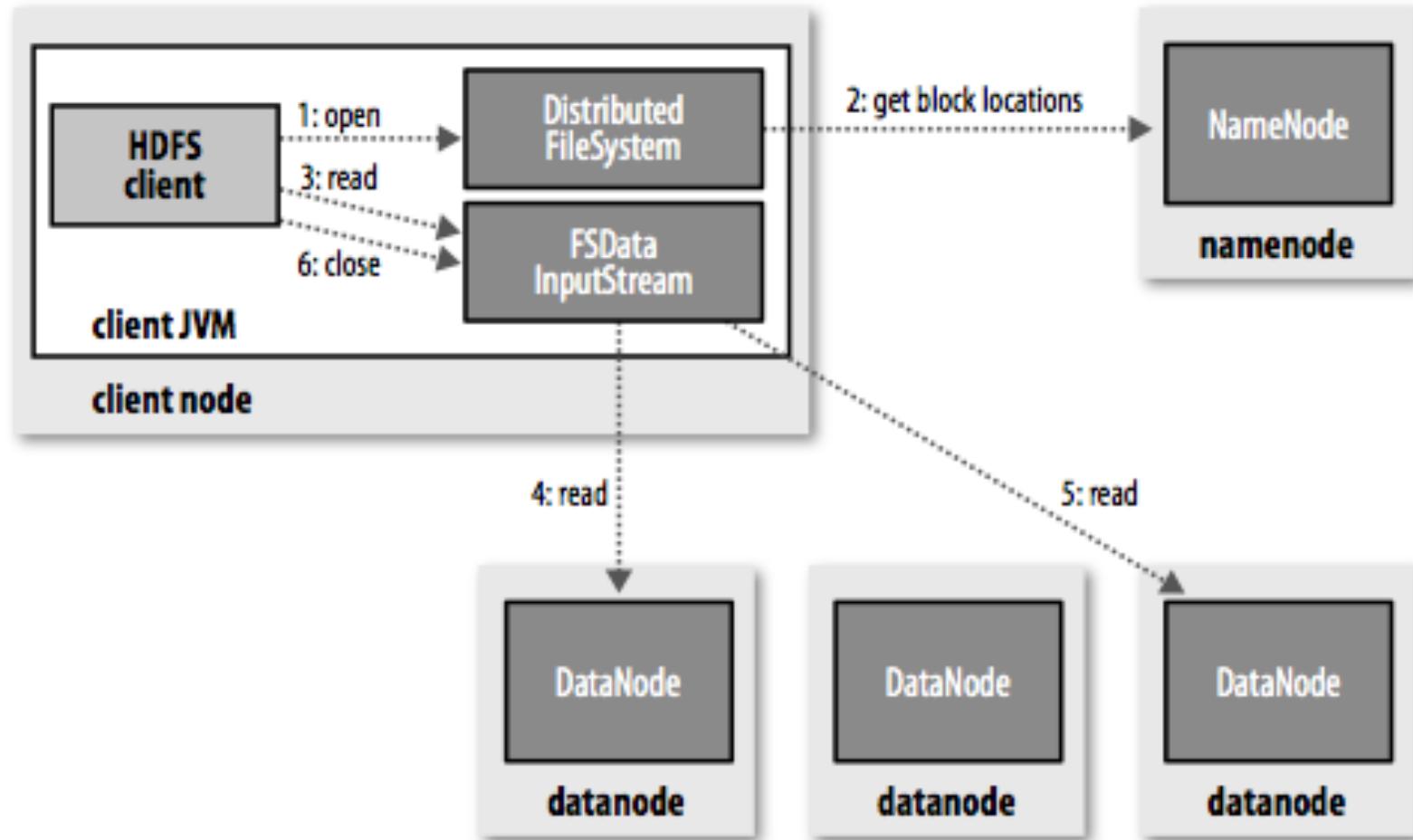
Hadoop is ‘Rack-aware’

- **Hadoop understands the concept of ‘rack awareness’**
 - The idea of where nodes are located, relative to one another
 - Helps the ResourceManager allocate processing resources on nodes closest to the data
 - Helps the NameNode determine the ‘closest’ block to a client during reads
 - In reality, this should perhaps be described as being ‘switch-aware’
- **HDFS replicates data blocks on nodes on different racks**
 - Provides extra data security in case of catastrophic hardware failure
- **Rack-awareness is determined by a user-defined script**
 - Rack Topology configuration details through Cloudera Manager discussed later in this course

HDFS Block Replication Strategy

- **First copy of the block is placed on the same node as the client**
 - If the client is not part of the cluster, the first block is placed on a random node
 - System tries to find one which is not too busy
- **Second copy of the block is placed on a node residing on a different rack**
- **Third copy of the block is placed on different node in the same rack as the second copy**

Anatomy of a File Read (1)



Anatomy of a File Read (2)

- 1. Client connects to the NameNode**
- 2. NameNode returns the name and locations of the first few blocks of the file**
 - Block locations are returned closest-first
- 3. Client connects to the first of the DataNodes, and reads the block**
 - If the DataNode fails during the read, the client will seamlessly connect to the next one in the list to read the block

Dealing With Data Corruption

- **As a client is reading the block, it also verifies the checksum**
 - ‘Live’ checksum is compared to the checksum created when the block was stored
- **If they differ, the client reads from the next DataNode in the list**
 - The NameNode is informed that a corrupted version of the block has been found
 - The NameNode will then re-replicate that block elsewhere
- **The DataNode verifies the checksums for blocks on a regular basis to avoid ‘bit rot’**
 - Default is every three weeks after the block was created

Data Reliability and Recovery

- **DataNodes send *heartbeats* to the NameNode**
 - Every three seconds
- **After a period without any heartbeats, a DataNode is assumed to be lost**
 - NameNode determines which blocks were on the lost node
 - NameNode finds other DataNodes with copies of these blocks
 - These DataNodes are instructed to copy the blocks to other nodes
 - Three-fold replication is actively maintained
- **A DataNode can rejoin a cluster after being down for a period**
 - The NameNode will ensure that blocks are not over-replicated by instructing DataNodes to remove excess copies
 - Note that this does not mean all blocks will be removed from the DataNode which was temporarily lost!

The NameNode Is Not a Bottleneck

- **Note: the data *never* travels via a NameNode**
 - For writes
 - For reads
 - During re-replication

Chapter Topics

HDFS

- HDFS Features
- Writing and Reading Files
- **NameNode Memory Considerations**
- Overview of HDFS Security
- Web UIs for HDFS
- Using the Hadoop File Shell
- Conclusion
- Hands-On Exercise: Working with HDFS

NameNode: Memory Allocation (1)

- When a NameNode is running, all metadata is held in RAM for fast response
- Default Java Heap Size of the NameNode is 1GB
 - At least 1GB recommended for every million HDFS blocks
- Items stored by the NameNode:
 - Filename, permissions, etc.
 - Block information for each block

NameNode: Memory Allocation (2)

- Why HDFS prefers fewer, larger files:

- Consider 1GB of data, HDFS block size 128MB
 - Stored as 1 x 1GB file
 - Name: 1 item
 - Blocks: 8 items
 - Total items in memory: 9
 - Stored as 1000 x 1MB files
 - Names: 1000 items
 - Blocks: 1000 items
 - Total items in memory: 2000

Chapter Topics

HDFS

- HDFS Features
- Writing and Reading Files
- NameNode Memory Considerations
- **Overview of HDFS Security**
- Web UIs for HDFS
- Using the Hadoop File Shell
- Conclusion
- Hands-On Exercise: Working with HDFS

HDFS File Permissions

- **Files in HDFS have an owner, a group, and permissions**
 - Very similar to Unix file permissions
- **File permissions are read (r), write (w), and execute (x) for each of owner, group, and other**
 - x is ignored for files
 - For directories, x means that its children can be accessed
- **HDFS permissions are designed to stop good people doing foolish things**
 - Not to stop bad people doing bad things!
 - HDFS believes you are who you tell it you are

Hadoop Security Overview

■ Authentication

- Proving that a user or system is who he or she claims to be
- Hadoop can provide strong authentication control via Kerberos
 - Cloudera Manager simplifies Kerberos deployment
- Authentication via LDAP is available with Cloudera Enterprise

■ Authorization (access control)

- Allowing people or systems to do some things but not other things
- CDH has traditional POSIX-style permissions for files and directories
- Access Control Lists (ACLs) for HDFS
- Role-based access control provided with Apache Sentry

■ Data encryption

- OS filesystem-level, HDFS-level, and Network-level options

■ We will cover Hadoop security in more depth later

Chapter Topics

HDFS

- HDFS Features
- Writing and Reading Files
- NameNode Memory Considerations
- Overview of HDFS Security
- **Web UIs for HDFS**
- Using the Hadoop File Shell
- Conclusion
- Hands-On Exercise: Working with HDFS

Web UIs for HDFS

The NameNode Web UI on port 50070

The screenshot shows the NameNode Web UI interface. At the top, there is a navigation bar with tabs: Hadoop, Overview, Datanodes, Snapshot, Startup Progress, and Utilities. The 'Overview' tab is selected. Below the navigation bar, the title 'Overview 'elephant:8020' (active)' is displayed. On the right side of the screen, there is a sidebar with various status indicators and logs. The main content area shows a file browser interface for the directory '/user/training/counts'. It lists several files and their details, such as mode (-rw-r--r--), owner (training), group (supergroup), and modification time (January 13 2015 5:37 PM). There are also sections for Quota Management and Snapshots.

The Cloudera Manager – HDFS File Browser

The screenshot shows the Cloudera Manager HDFS File Browser interface. At the top, there is a header with the Cloudera Manager logo, a search bar, and user information (Support, admin). Below the header, there is a navigation menu with links: Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, Administration, and a cluster dropdown for Cluster 1. The 'HDFS' link is highlighted. The main content area is titled 'File Browser' and shows a file listing for the directory '/user/training/counts'. The table includes columns for Name and Mode. To the right of the table, there is a detailed view of the directory '/user/training/counts' with information about its parent, owner, group, mode, modified time, and total objects. There are also buttons for Quota Management, Edit Quota, and Snapshots.

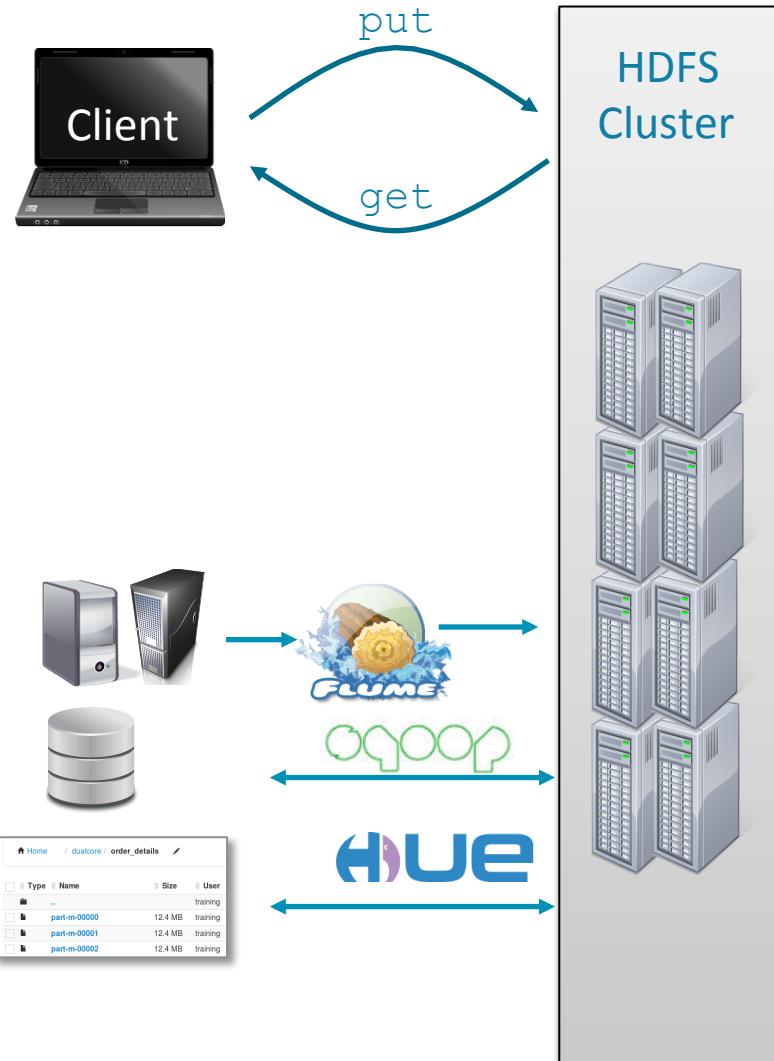
Chapter Topics

HDFS

- HDFS Features
- Writing and Reading Files
- NameNode Memory Considerations
- Overview of HDFS Security
- Web UIs for HDFS
- **Using the Hadoop File Shell**
- Conclusion
- Hands-On Exercise: Working with HDFS

Options for Accessing HDFS

- From the command line
 - FsShell: `hdfs dfs`
- From Cloudera Manager
 - HDFS page, File Browser tab
`hdfs://host:port/file...`
- From the NameNode Web UI
 - Utilities > Browse the file system
- Other programs
 - Java API
 - Used by MapReduce, Spark, Impala, Hue, Sqoop, Flume, etc.
 - RESTful interface



Accessing HDFS via the Command Line

- **HDFS is not a general purpose filesystem**
 - Not built into the OS, so only specialized tools can access it
- **End users typically access HDFS via the `hdfs dfs` command**
 - Actions are specified with subcommands (prefixed with a minus sign)
 - Most subcommands are similar to corresponding UNIX commands
- **Display the contents of the `/user/fred/sales.txt` file**

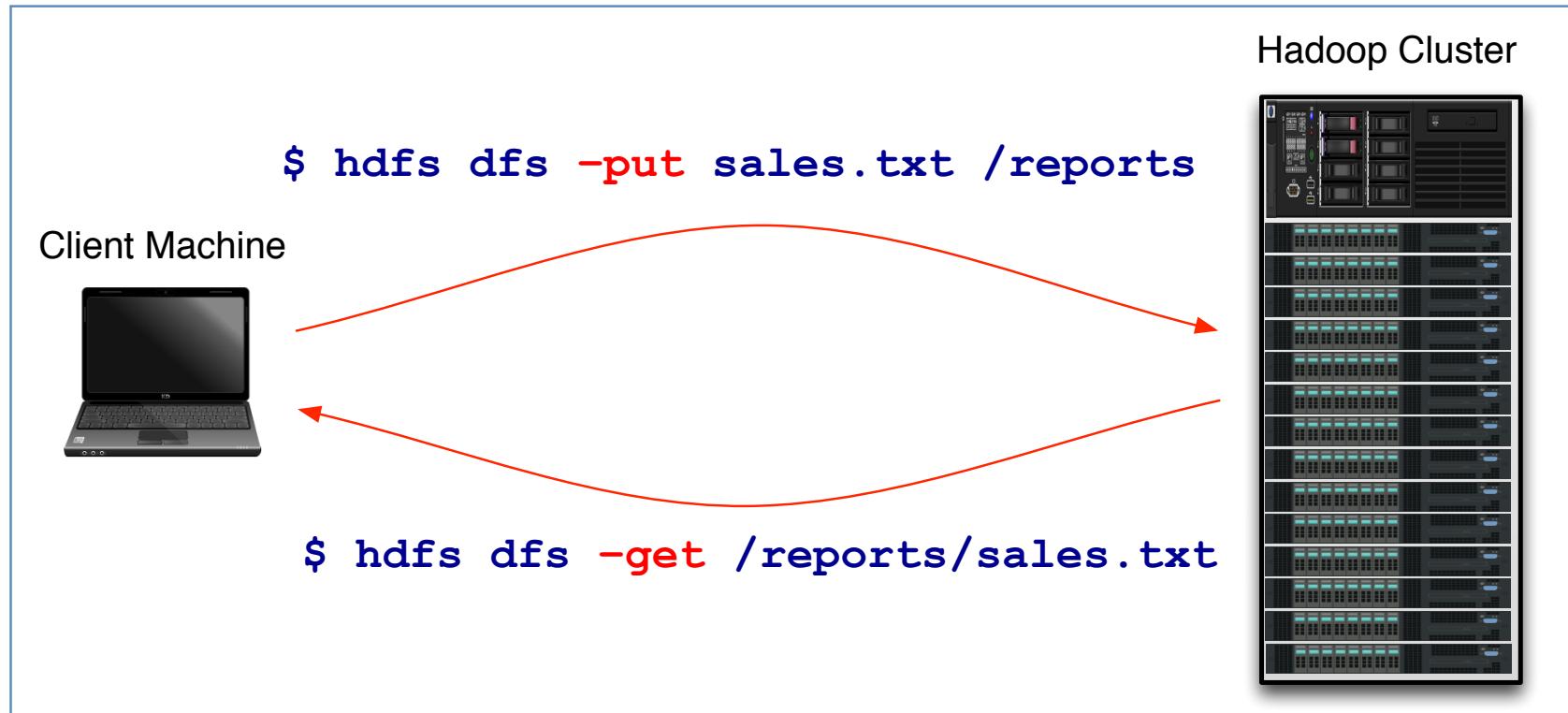
```
$ hdfs dfs -cat /user/fred/sales.txt
```

- **Create a directory (below the root) called `reports`**

```
$ hdfs dfs -mkdir /reports
```

Copying Local Data To and From HDFS Using the Command Line

- Remember that HDFS is distinct from your local filesystem
 - The hdfs dfs -put command copies local files **to** HDFS
 - The hdfs dfs -get fetches a local copy of a file **from** HDFS



More hdfs dfs Command Examples

- Copy file `input.txt` from local disk to the user's directory in HDFS

```
$ hdfs dfs -put input.txt input.txt
```

- This will copy the file to `/user/username/input.txt`
- Get a directory listing of the HDFS root directory

```
$ hdfs dfs -ls /
```

- Delete the file `/reports/sales.txt`

```
$ hdfs dfs -rm /reports/sales.txt
```

Chapter Topics

HDFS

- HDFS Features
- Writing and Reading Files
- NameNode Memory Considerations
- Overview of HDFS Security
- Web UIs for HDFS
- Using the Hadoop File Shell
- **Conclusion**
- Hands-On Exercise: Working with HDFS

Essential Points

- **HDFS distributes large blocks of data across a set of machines to support bringing the computation to the data**
 - Assumption is that the typical file size on a Hadoop cluster is large
 - Block size is configurable, default is 128MB
- **HDFS provides fault tolerance with built-in data redundancy**
 - The number of replicas per block is configurable, default is 3
- **The NameNode daemon maintains all HDFS metadata in memory and stores it on disk**
 - In a non-HA configuration, the NameNode works with a SecondaryNameNode that provides administrative services
 - In an HA configuration, the Standby NameNode is configured in case of a NameNode failure

Chapter Topics

HDFS

- HDFS Features
- Writing and Reading Files
- NameNode Memory Considerations
- Overview of HDFS Security
- Web UIs for HDFS
- Using the Hadoop file shell
- Conclusion
- **Hands-On Exercise: Working with HDFS**

Hands-On Exercise: Working With HDFS

- In this Hands-On Exercise, you will copy a large file into HDFS and explore the results in the NameNode Web UI, in Cloudera Manager, and in Linux
- Please refer to the Hands-On Exercise Manual

MapReduce and Spark on YARN

Chapter 5



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- **MapReduce and Spark on YARN**
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

MapReduce and Spark on YARN

In this chapter, you will learn:

- **The role of computational frameworks in Hadoop**
- **The purpose and basic architecture of YARN**
- **MapReduce Concepts**
- **Apache Spark Concepts**
- **How YARN allocates cluster resources**
- **How YARN handles failure**
- **How to Review and Manage YARN applications**
- **How to access YARN application logs**

Chapter Topics

MapReduce and Spark on YARN

- **The Role of Computational Frameworks**
- YARN: The Cluster Resource Manager
- MapReduce Concepts
- Apache Spark Concepts
- Running Computational Frameworks on YARN
- Exploring YARN Applications Through the Web UIs, and the Shell
- YARN Application Logs
- Conclusion
- Hands-On Exercise: Running YARN Applications

Hadoop Computational Frameworks

- **HDFS provides scalable storage in your Hadoop cluster**
- **Computational frameworks provide the distributed computing**
 - Batch processing
 - SQL queries
 - Search
 - Machine learning
 - Stream processing
- **Computational frameworks compete for resources**
- **YARN provides resource management for computational frameworks that support it**
 - Examples discussed in this chapter:
 - MapReduce
 - Apache Spark

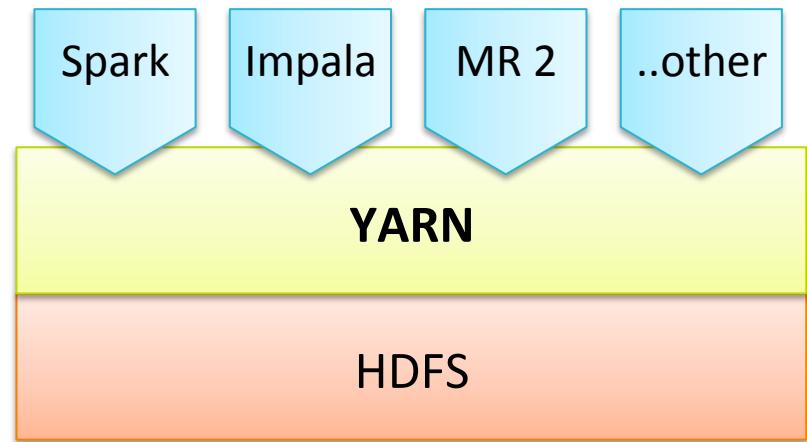
Chapter Topics

MapReduce and Spark on YARN

- The Role of Computational Frameworks
- **YARN: The Cluster Resource Manager**
- MapReduce Concepts
- Apache Spark Concepts
- Running Computational Frameworks on YARN
- Exploring YARN Applications Through the Web UIs, and the Shell
- YARN Application Logs
- Conclusion
- Hands-On Exercise: Running YARN Applications

What is YARN?

- Yet Another Resource Negotiator (YARN)
- A platform for managing resources in a Hadoop cluster
- Supports a growing number of Hadoop distributed processing frameworks, including:
 - MapReduce v2
 - Spark
 - Impala
 - Others



Why YARN? (1)

- **YARN allows you to run diverse workloads on the same Hadoop cluster**
 - Jobs using different frameworks will probably have different resource profiles
- **Examples:**
 - A MapReduce job or an Impala query that scans a large table
 - Likely heavily disk-bound
 - Requires little memory
 - A Spark job executing an iterative machine learning algorithm
 - Will probably attempt to store the entire dataset in memory
 - May use spurts of CPU to perform complex computations
- **YARN allows you to share cluster memory and CPU resources dynamically between processing frameworks**
 - MapReduce, Impala, Spark, and others

Why YARN? (2)

- **Achieve more predictable performance**
 - Avoid ‘oversubscribing’ nodes
 - Requesting more processing power or RAM than is available
 - Protect higher-priority workloads with better isolation
- **Increase cluster utilization**
 - Resource needs and capacities can be configured less conservatively than would otherwise be possible

Notable Computational Frameworks on YARN

- **MapReduce**

- The original framework for writing Hadoop applications
- Proven, widely used
- Sqoop, Hive, Pig, other tools use MapReduce to interact with HDFS

- **Spark**

- A newer programming framework for writing Hadoop applications
- Production-ready
- Supports processing of streaming data
- Faster than MapReduce

Chapter Topics

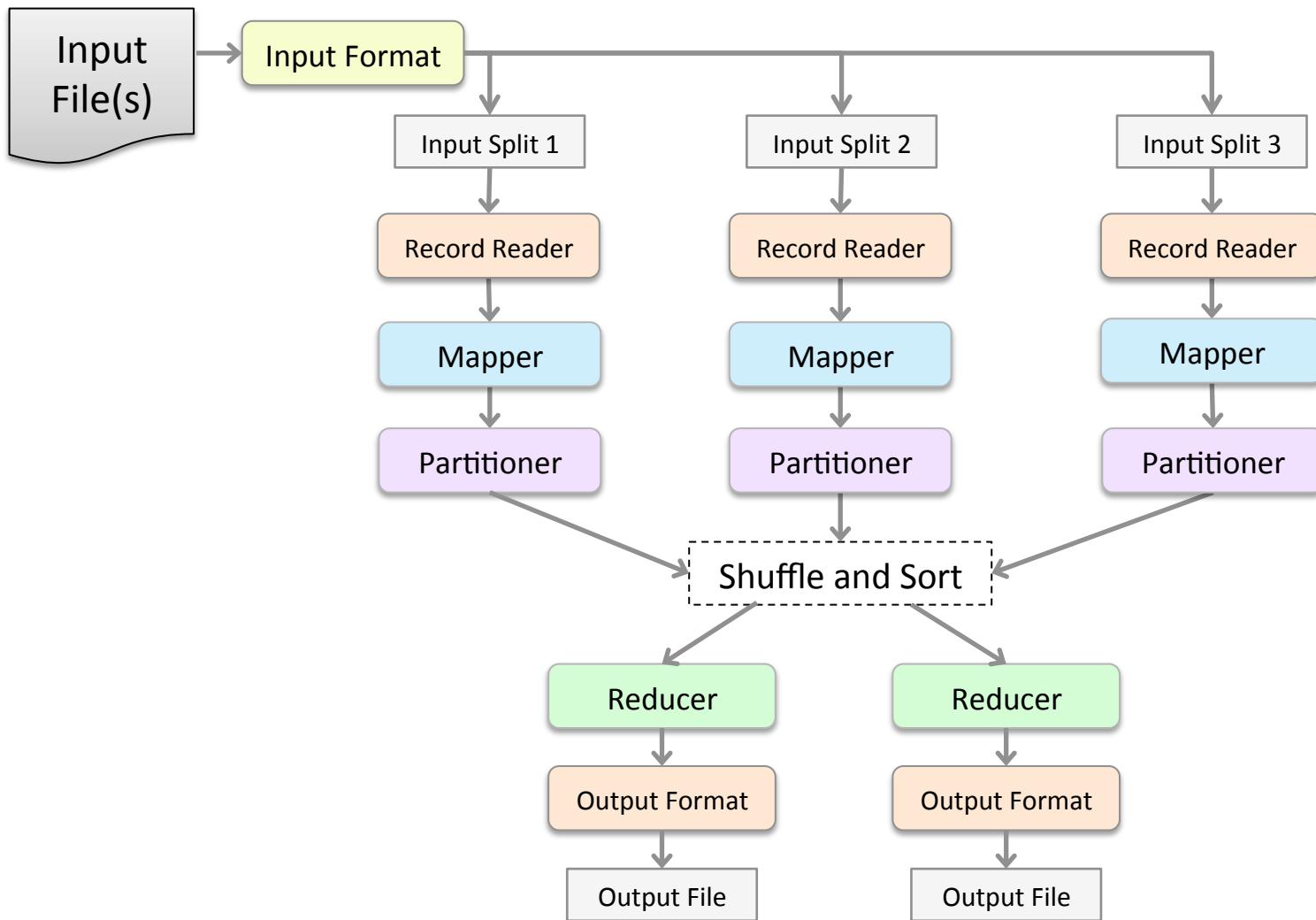
MapReduce and Spark on YARN

- The Role of Computational Frameworks
- YARN: The Cluster Resource Manager
- **MapReduce Concepts**
- Apache Spark Concepts
- Running Computational Frameworks on YARN
- Exploring YARN Applications Through the Web UIs, and the Shell
- YARN Application Logs
- Conclusion
- Hands-On Exercise: Running YARN Applications

What Is MapReduce?

- **MapReduce is a programming model**
 - Neither platform- nor language-specific
 - Record-oriented data processing (key and value)
 - Facilitates task distribution across multiple nodes
- **Where possible, each node processes data stored on that node**
- **Consists of two developer-created phases**
 - Map
 - Reduce
- **In between Map and Reduce is the *shuffle and sort***
 - Sends data from the Mappers to the Reducers

MapReduce: The Big Picture



Features of MapReduce

- **Automatic parallelization and distribution**
- **Fault-tolerance**
- **Status and monitoring tools**
- **A clean abstraction for programmers**
 - MapReduce programs are usually written in Java
 - Can be written in other languages using *Hadoop Streaming*
- **MapReduce abstracts all the ‘housekeeping’ away from the developer**
 - Developer can concentrate simply on writing the Map and Reduce functions

MapReduce Concepts

- Each Mapper processes a single *input split* from HDFS
 - Often a single HDFS block
- Hadoop passes one *record* at a time to the developer's Mapper code
- Each record has a *key* and a *value*
- *Intermediate data* is written by the Mapper to local disk
- During the shuffle and sort phase, all the values associated with the same intermediate key are transferred to the same Reducer
 - The developer specifies the number of Reducers
- Reducer is passed each key and a list of all its values
 - Keys are passed in sorted order
- Output from the Reducers is written to HDFS

MapReduce Application Terminology

- **Job**

- A Mapper, a Reducer, and a list of inputs to process
 - The highest-level unit of computation in MapReduce

- **Task**

- An individual unit of work
 - A job consists one or more tasks
 - Each task is either a Map task or a Reduce task
 - Runs in a container on a worker node

- **Client**

- The program that submits the job to the YARN ResourceManager
 - Sometimes refers to the machine on which the program runs

Deploying MapReduce

- **MapReduce is part of CDH 5**
 - With Cloudera Manager: Use the ‘Add a Service’ wizard to add YARN (MR2 Included) to your cluster
- **MapReduce Service – two versions available**
 - MapReduce v2
 - Recommended
 - MapReduce applications run as YARN applications
 - MapReduce v1
 - For backward compatibility
 - Does not use YARN
 - Both versions require HDFS

Chapter Topics

MapReduce and Spark on YARN

- The Role of Computational Frameworks
- YARN: The Cluster Resource Manager
- MapReduce Concepts
- **Apache Spark Concepts**
- Running Computational Frameworks on YARN
- Exploring YARN Applications Through the Web UIs, and the Shell
- YARN Application Logs
- Conclusion
- Hands-On Exercise: Running YARN Applications

What is Apache Spark?

- **A fast, general engine for large-scale data processing on a cluster**
 - One of the fastest-growing Apache projects
 - Includes map and reduce as well as non-batch processing models
- **High-level programming framework**
 - Programmers can focus on logic not plumbing
 - Works directly with HDFS
 - Near real-time processing
 - Configurable in-memory data caching for efficient iteration
- **Application processing is distributed across worker nodes**
 - Distributed storage, horizontally scalable, fault tolerance



Spark Applications

- **Spark Shell**

- Interactive: for learning, exploring data
 - Python or Scala

- **Spark Applications**

- Support large scale data processing
 - Python, Scala, or Java
 - A Spark Application consists of one or more jobs
 - A job consists of one or more tasks

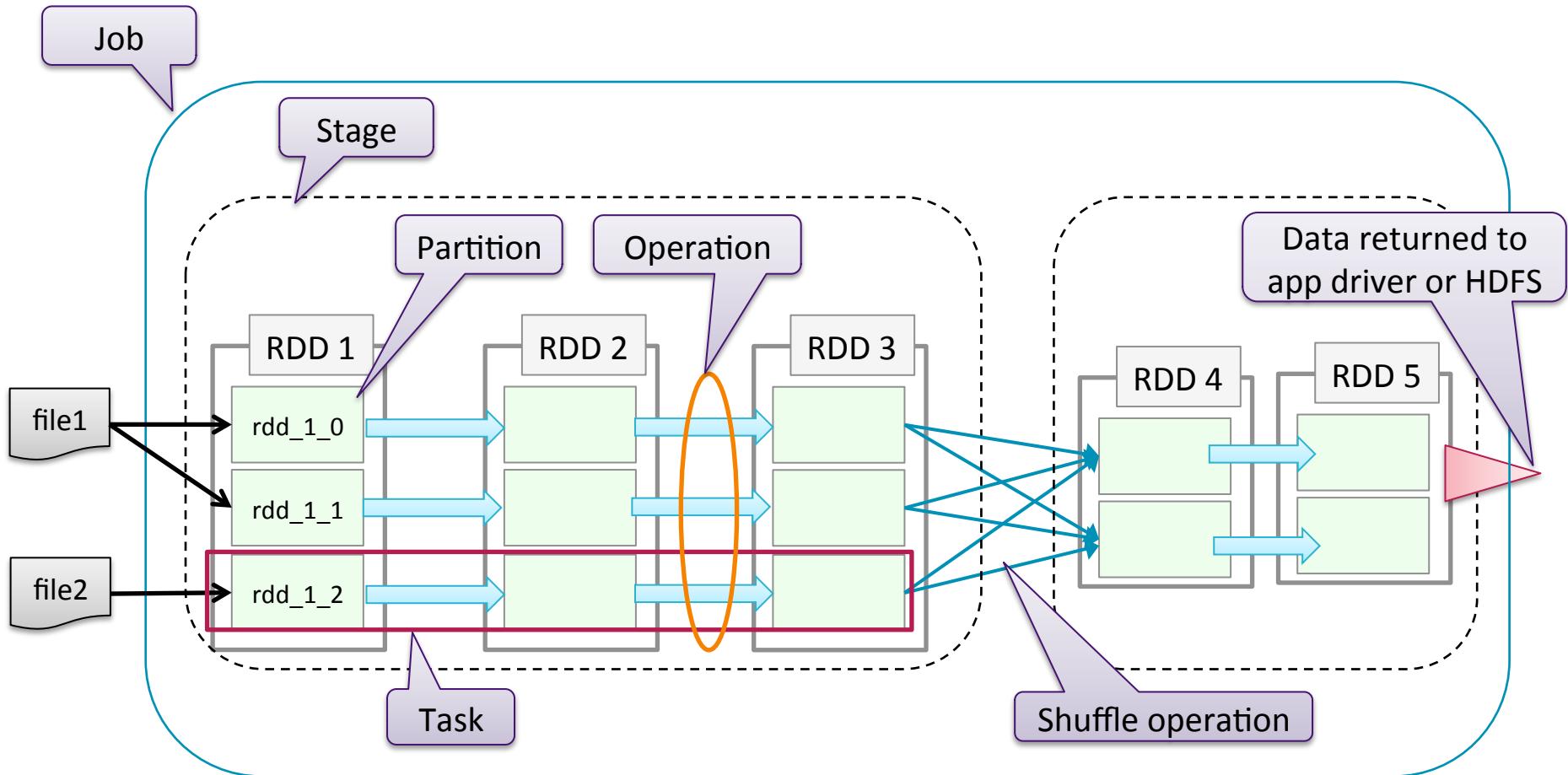
- **Every Spark application has a Spark Driver**

- In “yarn-client” mode, the driver runs on the client
 - In “yarn-cluster” mode, the driver runs on the cluster, on the ApplicationMaster
 - In this case, if the client disconnects the application will continue to run

Spark RDDs

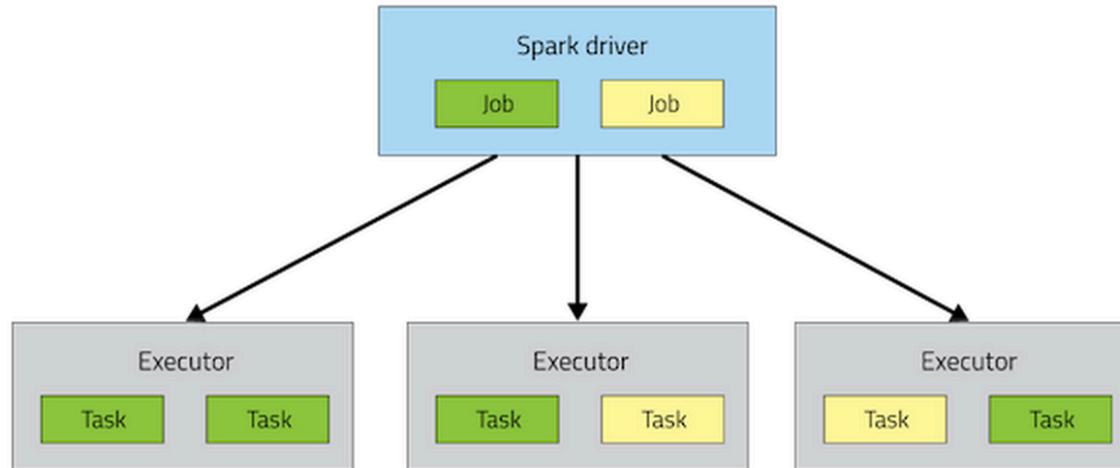
- **RDD- the fundamental unit of data in Spark**
 - Resilient – if data in memory is lost, it can be recreated
 - Distributed – stored in memory across the cluster
 - Dataset – initial data comes from a file or is created programmatically
- **RDDs are created from file(s), data in memory or another RDD**
- **Most Spark programming consists of performing operations on RDDs**
 - Two types of operations
 - **Transformations:** create a new RDD based on an existing one
 - **Actions:** return a value from an RDD

Spark: The Big Picture



Spark Executors

- A Spark Application consists of one or more Spark jobs
 - Jobs run tasks in executors
 - Executors run in YARN containers (one executor per container)
 - Executors exist for the life of the application
 - The number of executors for an application is fixed once it starts
 - YARN does not yet support already allocated container resizing
 - An executor can run tasks concurrently on in-memory data



Deploying Spark

- **Spark is part of CDH 5**

- With Cloudera Manager: ‘Add a Service’ to your cluster

- **Spark Service – two versions available**

- Spark on YARN - recommended
 - Spark applications run as YARN applications
 - Requires HDFS and YARN
 - Supported with Cloudera Manager 5.2 and above
 - Spark (standalone)
 - Both versions come with a Spark History Server

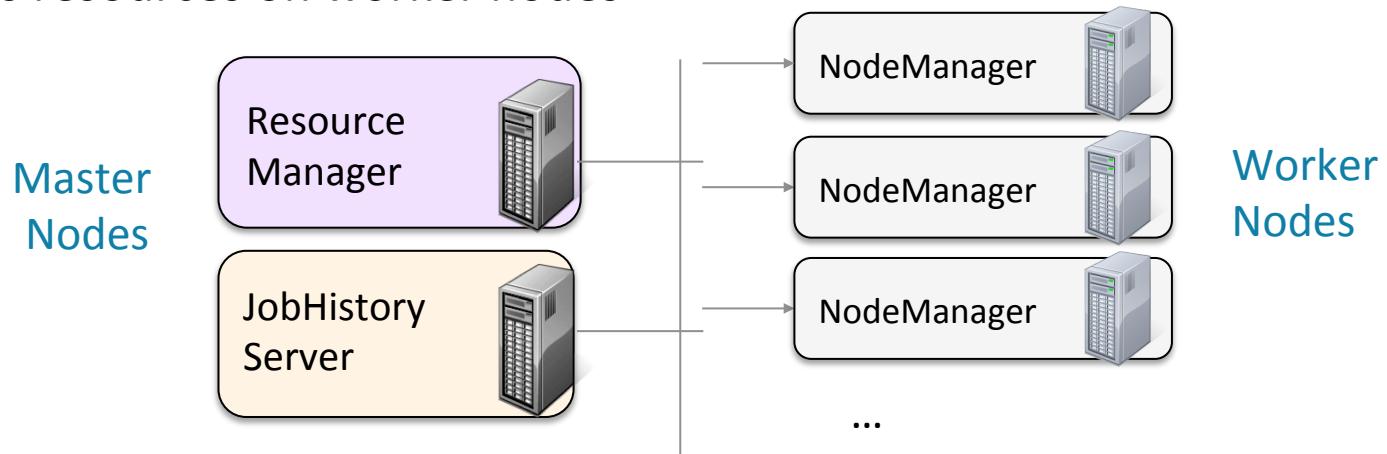
Chapter Topics

MapReduce and Spark on YARN

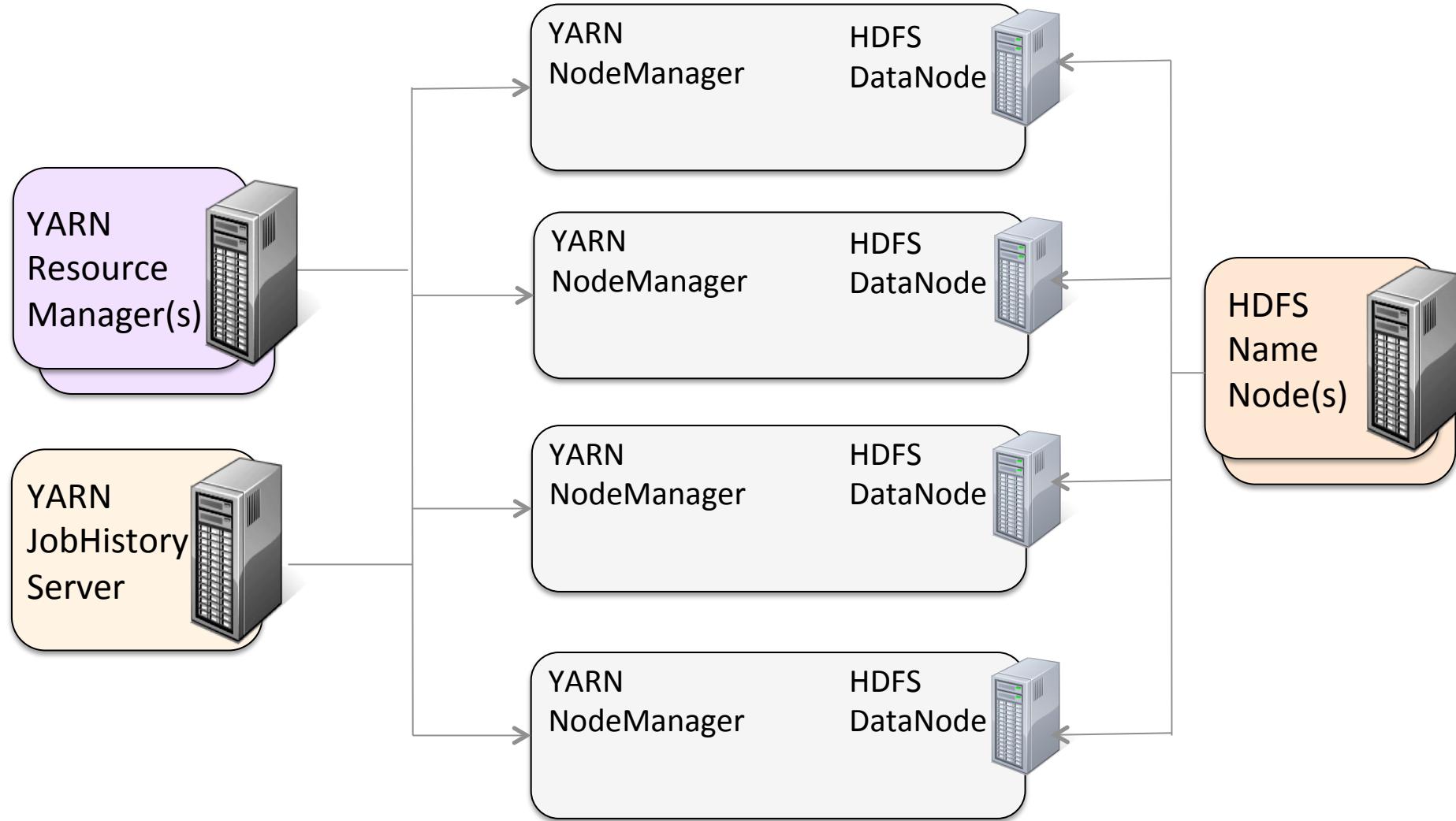
- The Role of Computational Frameworks
- YARN: The Cluster Resource Manager
- MapReduce Concepts
- Apache Spark Concepts
- **Running Computational Frameworks on YARN**
- Exploring YARN Applications Through the Web UIs, and the Shell
- YARN Application Logs
- Conclusion
- Hands-On Exercise: Running YARN Applications

YARN Daemons

- **ResourceManager – one per cluster**
 - Initiates application startup
 - Schedules resource usage on worker nodes
- **JobHistoryServer – one per cluster**
 - Archives MapReduce jobs' metrics and metadata
- **NodeManager – one per worker node**
 - Starts application processes
 - Manages resources on worker nodes



A Typical YARN Cluster



YARN ResourceManager – Key Points

- **What the ResourceManager does:**

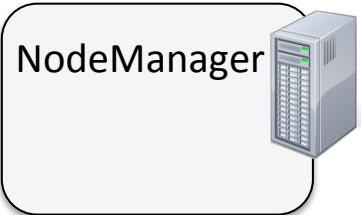
- Manages nodes
 - Tracks heartbeats from NodeManagers
- Runs a scheduler
 - Determines how resources are allocated
- Manages containers
 - Handles ApplicationMasters' requests for resources
 - Deallocates containers when they expire or when the application completes
- Manages ApplicationMasters
 - Creates a container for ApplicationMasters and tracks heartbeats
- Manages cluster-level security



YARN NodeManagers – Key Points

- **What NodeManagers do:**

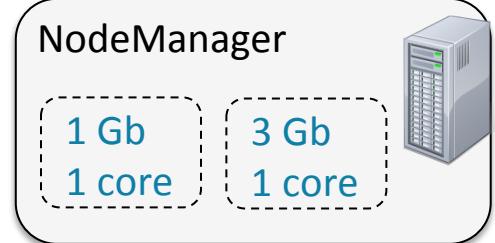
- Communicate with the ResourceManager
 - Register and provide info on node resources
 - Send heartbeats and container status
- Manage processes in containers
 - Launch ApplicationMasters on request from the ResourceManager
 - Launch processes into containers on request from ApplicationMasters
 - Monitor resource usage by containers; kill runaway processes
- Provide logging services to applications
 - Aggregate logs for an application and save them to HDFS
- Run auxiliary services
- Maintain node level security



Running an Application in YARN

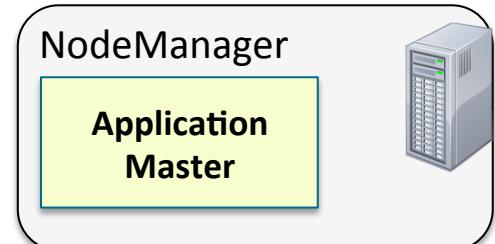
■ Containers

- Allocated by the ResourceManager
- Require a certain amount of resources (memory, CPU) on a worker node
- YARN Applications run in one or more containers



■ Application Master

- One per YARN application
- Runs in a container
- Framework/application specific
- Communicates with the ResourceManager scheduler to request containers to run application tasks
- Ensures NodeManager(s) complete tasks



YARN: Submit a MapReduce Application

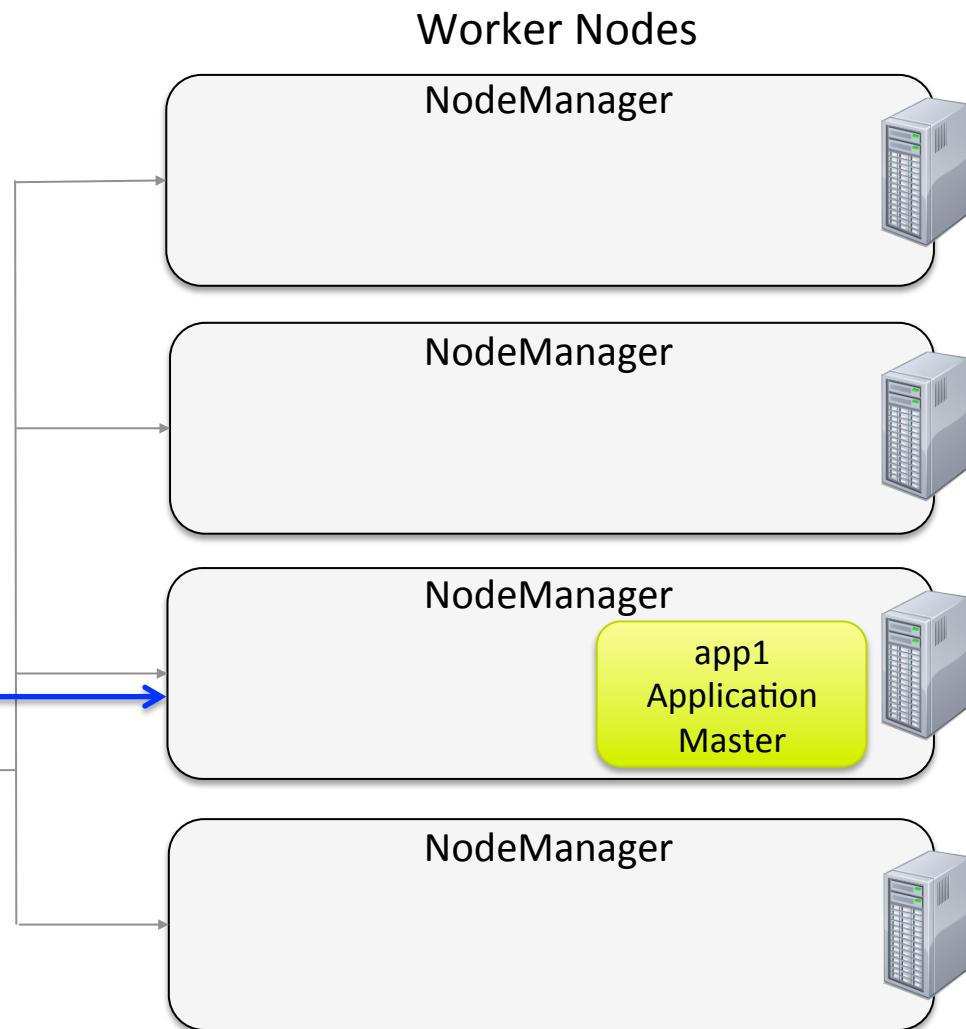
```
$ hadoop jar MyMR.jar  
MyDriver mydata outdir
```



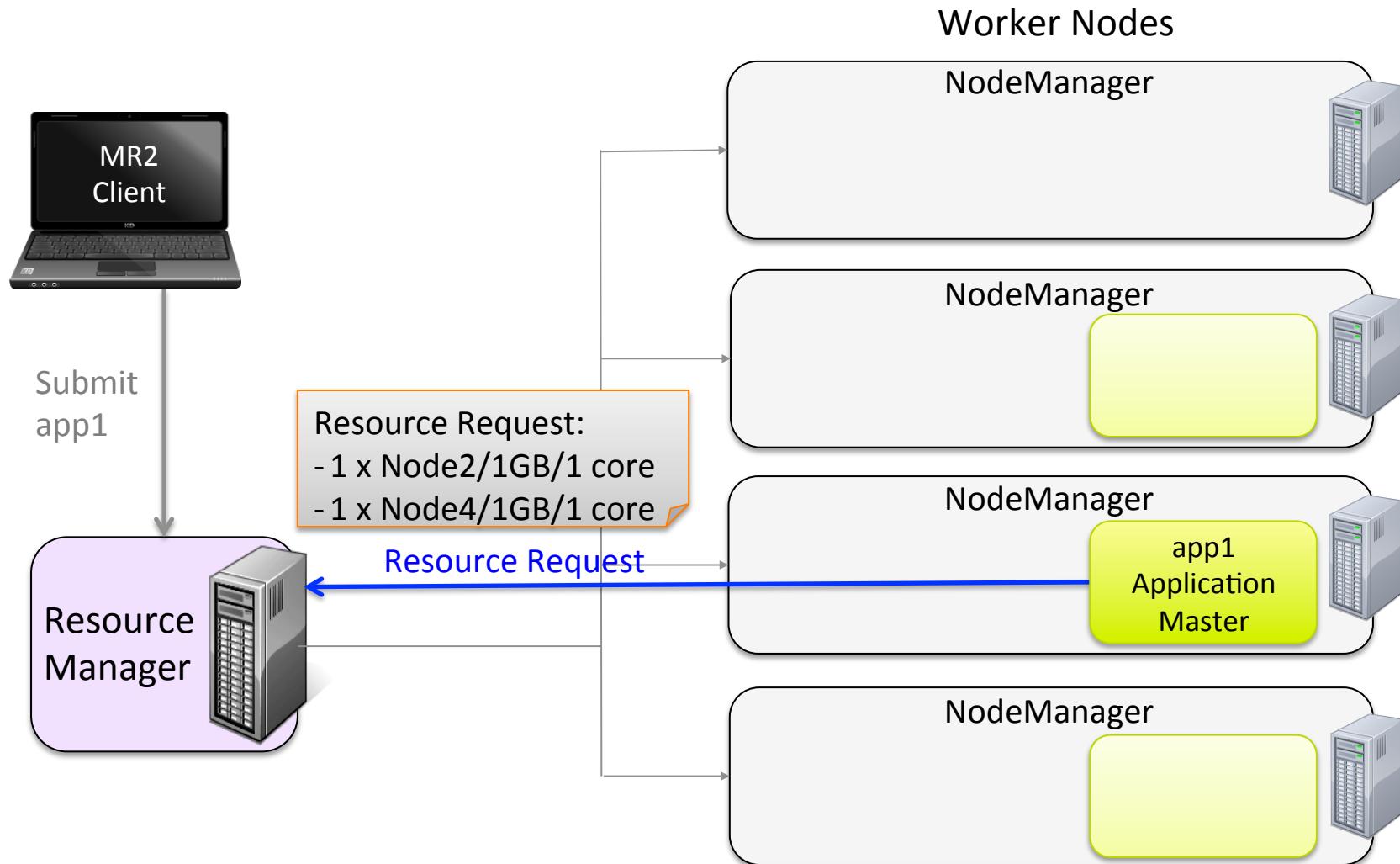
Submit
app1



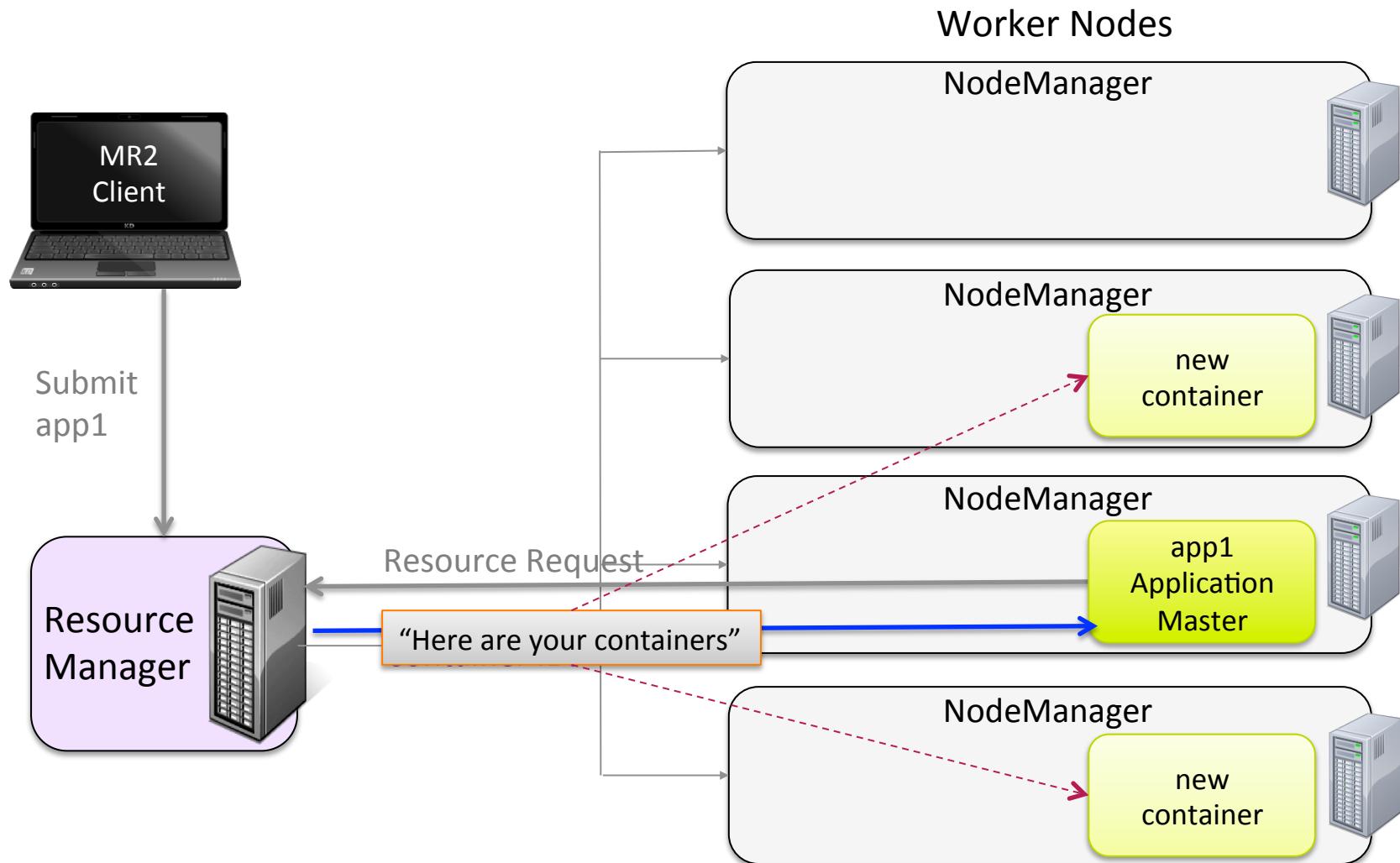
Launch app1



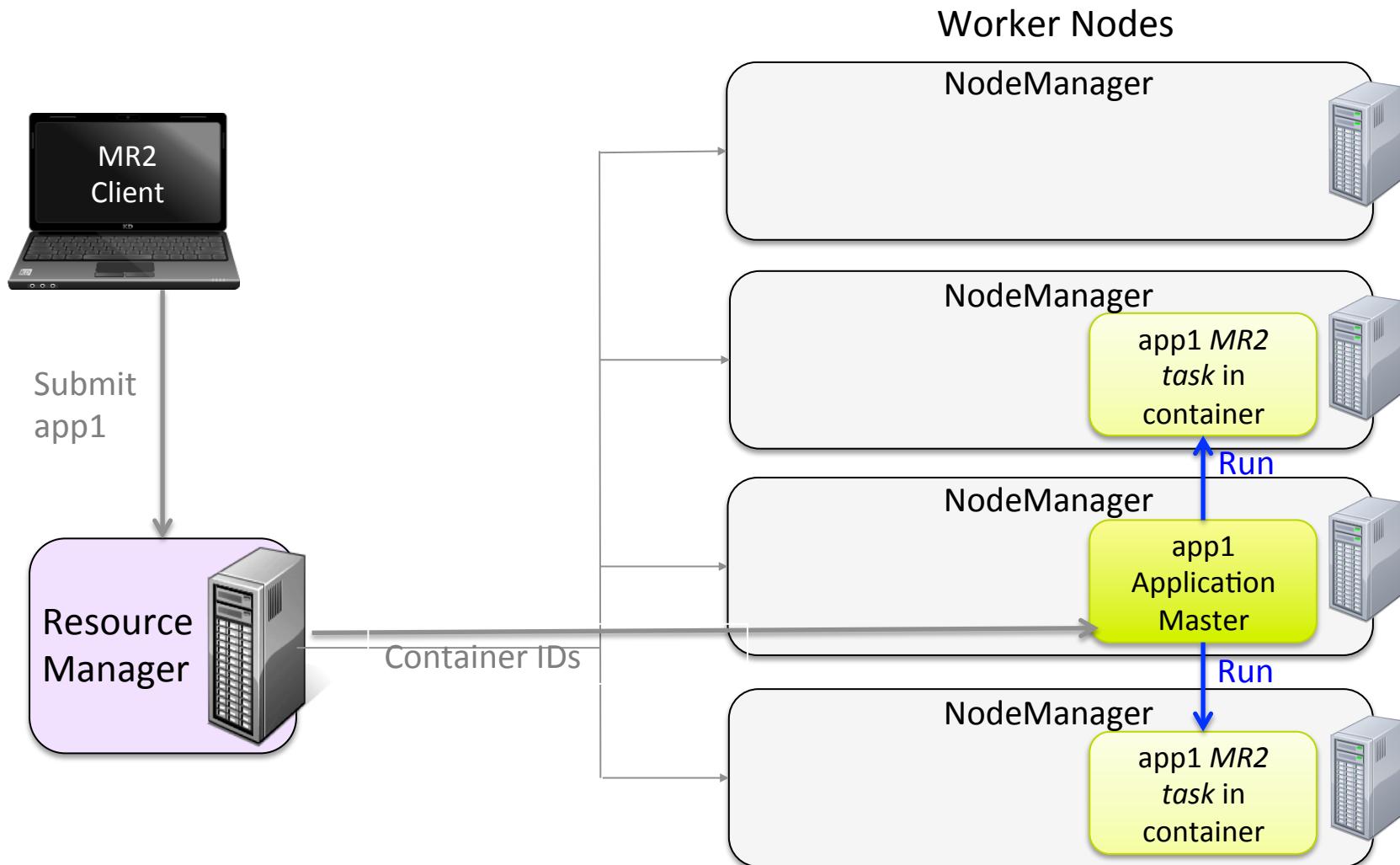
YARN: Application Master Resource Request



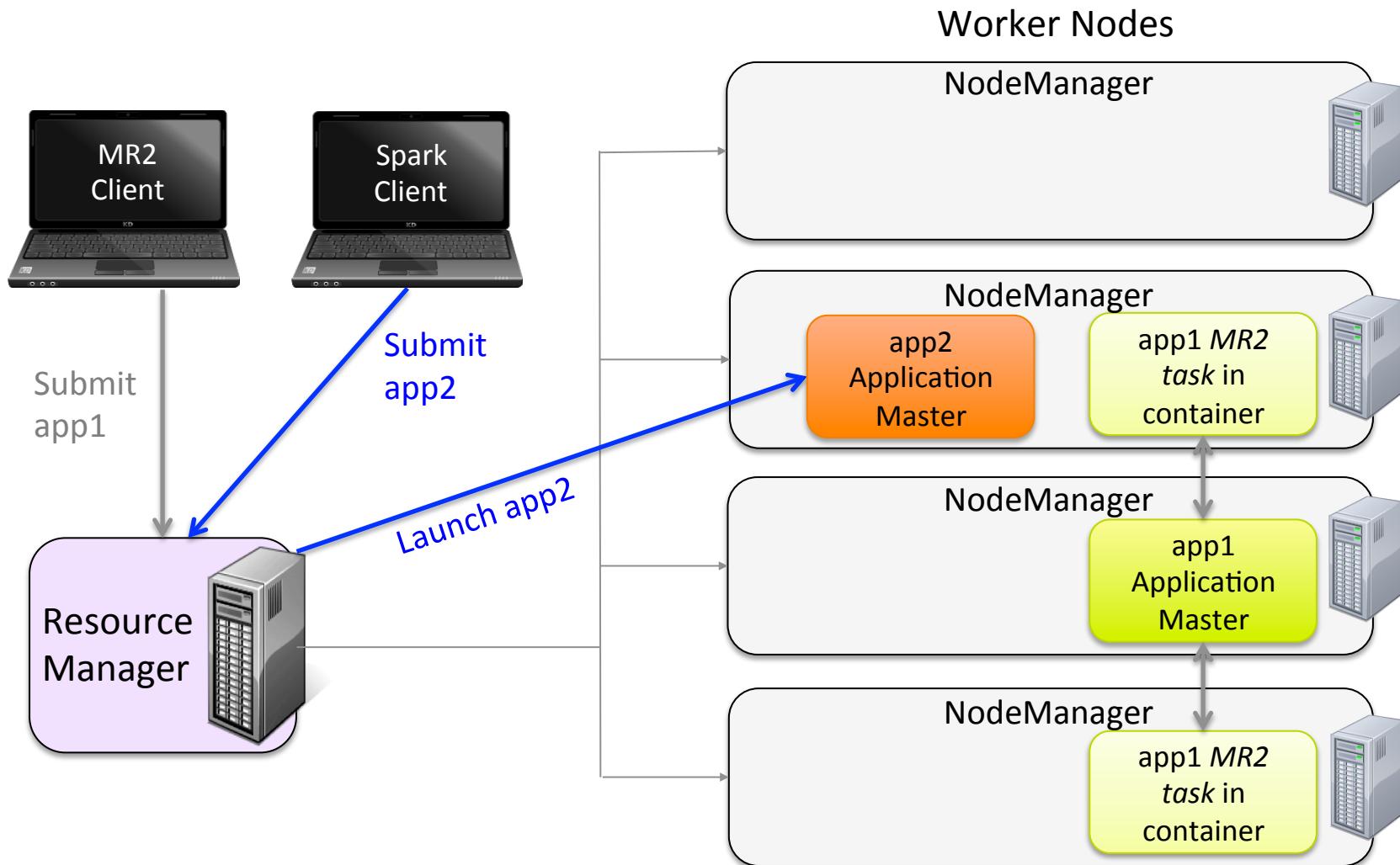
YARN: Container Allocation



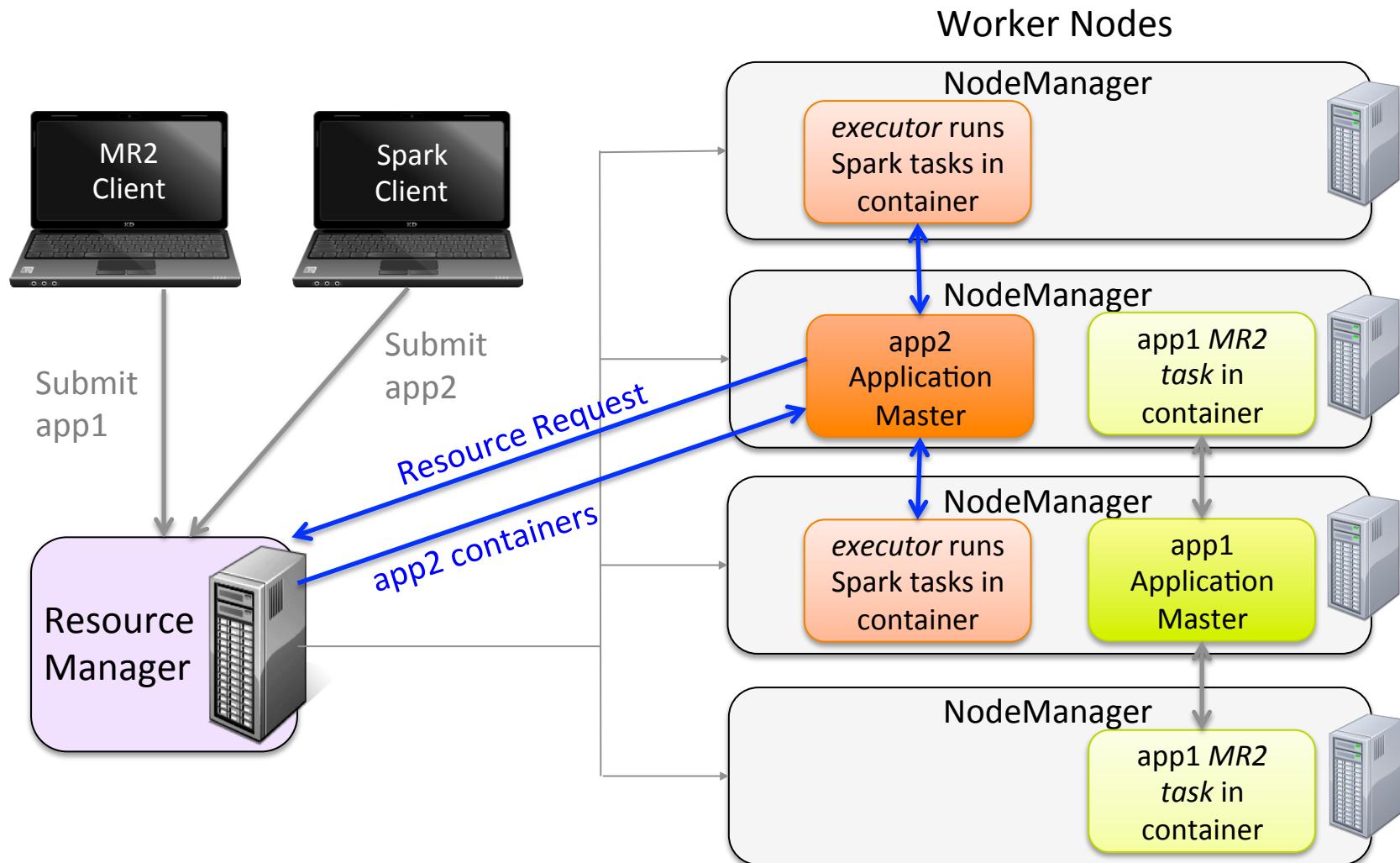
YARN: Application Containers



YARN: Submit a Spark Application



YARN: Containers



YARN Container Lifespan

- **MapReduce's use of containers**

- One container is requested and created *for each task* in a job
- Each Map or Reduce task gets its own JVM that runs in a container
- Each container is deleted once a task completes

- **Spark's use of containers**

- One container is requested and created *for each executor* granted to the Spark application
- An executor is a JVM
 - Many Spark tasks can run in a single executor – concurrently and over the lifetime of the container
- Each container stays alive for the lifespan of the application

YARN and Data Locality

- **The YARN Scheduler aims for data locality**
 - Objective: Bring the compute to the data
- **ApplicationMasters know where the HDFS blocks required to complete an application task are located**
 - ApplicationMasters inform the YARN scheduler which node is preferred to accomplish data locality
 - Challenge: Spark must ask YARN for executors *before* jobs are run
 - Application developer can inform YARN which files will be processed
- **When resource availability permits, the YARN Scheduler assigns containers to nodes closest to the data**
 - If the node is not available, YARN will prefer at least the same rack where the task input data resides
 - Scheduling is discussed in more detail later in this course

Summary: Cluster Resource Allocation

- **Resource Manager (master)**
 - Grants containers
 - Performs cluster scheduling
- **Application Master (runs within a container)**
 - Negotiates with the Resource Manager to obtain containers on behalf of the application
 - Presents containers to Node Managers
- **Node Managers (workers)**
 - Manage life-cycle of containers
 - Launch Map and Reduce tasks or Spark executors in containers
 - Monitor resource consumption

Summary: Requesting Resources

- A resource request is a fine-grained request for memory and CPU sent to the Resource Manager to be fulfilled
- If the request is successful, a container is granted
- A resource request is composed of several fields that specify
 - The amount of a given resource required
 - Data locality information, i.e., the preferred node or rack on which to run

Field Name	Sample Value
priority	integer
capability	<2 gb, 1 vcore>
resourceName	host22, Rack5, *
numContainers	integer

YARN Fault Tolerance (1)

Failure	Action Taken
ApplicationMaster stops sending heartbeats	ResourceManager reattempts the whole application (default: 2 times)
YARN application fails	ResourceManager reattempts the whole application (default: 2 times)
MR task exits with exceptions	ApplicationMaster reattempts the task in a new container on a different node (default: 4 times)
An MR task stops responding	ApplicationMaster reattempts the task in a new container on a different node (default: 4 times)
MR task fails too many times	Task aborted
Spark executor fails	Spark launches new executors (default: Spark tolerates 2 * number of requested executors failing before Spark fails the app)
A Spark task fails	Spark Task Scheduler resubmits task to run on different executor

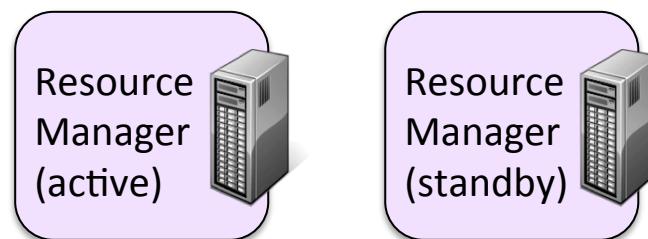
YARN Fault Tolerance (2)

■ NodeManager

- If the NodeManager stops sending heartbeats to the ResourceManager, it is removed from list of active nodes
- Tasks on the node will be treated as failed by the ApplicationMaster
- If the ApplicationMaster node fails, it will be treated as a failed application

■ ResourceManager

- No applications or tasks can be launched if the ResourceManager is unavailable
- Can be configured with high availability (HA)



Chapter Topics

MapReduce and Spark on YARN

- The Role of Computational Frameworks
- YARN: The Cluster Resource Manager
- MapReduce Concepts
- Apache Spark Concepts
- Running Computational Frameworks on YARN
- **Exploring YARN Applications Through the Web UIs, and the Shell**
- YARN Application Logs
- Conclusion
- Hands-On Exercise: Running YARN Applications

Cloudera Manager – YARN Applications UI

The screenshot shows the Cloudera Manager YARN Applications UI. At the top, there's a navigation bar with links for Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. Below that is a cluster navigation bar for Cluster 1, showing a date range from 7 days preceding December 23, 2014, at 11:29 AM PST. The main title is "YARN (MR2 Included)". The "Applications" tab is selected, showing a search bar and time filters (30m, 1h, 2h, 6h, 12h, 1d, 7d, 30d). A "Workload Summary" section for completed applications includes CPU Time and Duration filters. The "Results" section displays completed applications, with one entry highlighted: "Spark shell" with ID [application_1419030083499_0005](#). This entry has a red box around it and a blue arrow pointing to a purple callout box labeled "Link to YARN ResourceManager Web UI". Another application entry is shown below it: "Spark shell" with ID [application_1419030083499_0004](#). The "Charts" tab is also visible. On the right, there are "Actions" and "Export" buttons, with the "Export" button also having a red box around it.

cloudera manager

Home Clusters ▾ Hosts Diagnostics ▾ Audits Charts ▾ Backup ▾ Administration ▾

Cluster 1 < 7 days preceding December 23 2014, 11:29 AM PST > Actions ▾

YARN (MR2 Included)

Status Instances Configuration Commands Audits Applications Charts Library

Applications

30m 1h 2h 6h 12h 1d 7d 30d

Search for YARN applications, e.g. 'pool = default' or press space to start typeahead. ⋮

Search Select Attributes

Workload Summary

(For Completed Applications)

CPU Time

- 4.5s - 5.4s
- 8.1s - 9s

Duration

- 20s - 3.3m
- 3.3m - 33.3m
- 33.3m - 5.6h

Results Charts

12/19/2014 4:30 PM - 12/23/2014 10:46 AM

Spark shell
ID: [application_1419030083499_0005](#) Type: SPARK
User: training Pool: root.training
Duration: 3.8d

Spark shell
ID: [application_1419030083499_0004](#) Type: SPARK
User: training Pool: root.training
Duration: 26.32s

Link to YARN ResourceManager Web UI

Export

Other YARN Application Web UIs

■ ResourceManager Web UI

- Provides cluster utilization metrics on nodes, application status, application scheduling, and links to logs
- Embeds links to the UIs listed below

ID	User	Name	Application Type	Queue	StartTime	FinishTime
application_1384200217415_0009	training	Process Logs	MAPREDUCE	root.training	Tue, 12 Nov 2013 21:48:38 GMT	N/A
application_1384200217415_0008	training	Average Word Length	MAPREDUCE	root.training	Mon, 11 Nov 2 21:55:21 2013 GMT	Mon, 11 Nov 2 21:56:33 2013 GMT
application_1384200217415_0007	training	Process Logs	MAPREDUCE	root.training	Mon, 11 Nov 2013 21:48:38 GMT	Mon, 11 Nov 2 21:44:17 2013 GMT
application_1384200217415_0006	training	Process	MAPREDUCE	root.training	Mon, 11 Nov 2 21:48:38 GMT	Mon, 11 Nov 2 21:44:17 2013 GMT

<http://rmhost:8088>

■ MapReduce JobHistory Server Web UI

- http://jhs_host:19888
- Provides details on retired jobs including state, time metrics, Map and Reduce task details and logs

Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Completed	Re
2013.11.21 13:07:38	2013.11.21 13:08:27	job_1385066116114_0004	Process Logs	cloudera	default	SUCCEEDED	4	4	12
2013.11.21 13:03:53	2013.11.21 13:04:42	job_1385066116114_0003	Process Logs	cloudera	default	SUCCEEDED	4	4	12
2013.11.21 13:01:35	2013.11.21 13:02:28	job_1385066116114_0002	Process Logs	cloudera	default	SUCCEEDED	4	4	12
2013.11.21 12:48:00	2013.11.21 12:50:43	job_1385066116114_0001	Word Count	cloudera	default	SUCCEEDED	4	4	1
2013.11.21 09:24:45	2013.11.21 09:28:19	job_1385049040288_0003	Word Count	cloudera	default	SUCCEEDED	4	4	1

http://jhs_host:19888

■ Spark History Server Web UI

- http://shs_host:18088
- Provides details on Spark jobs, stages, storage, environment, and executors

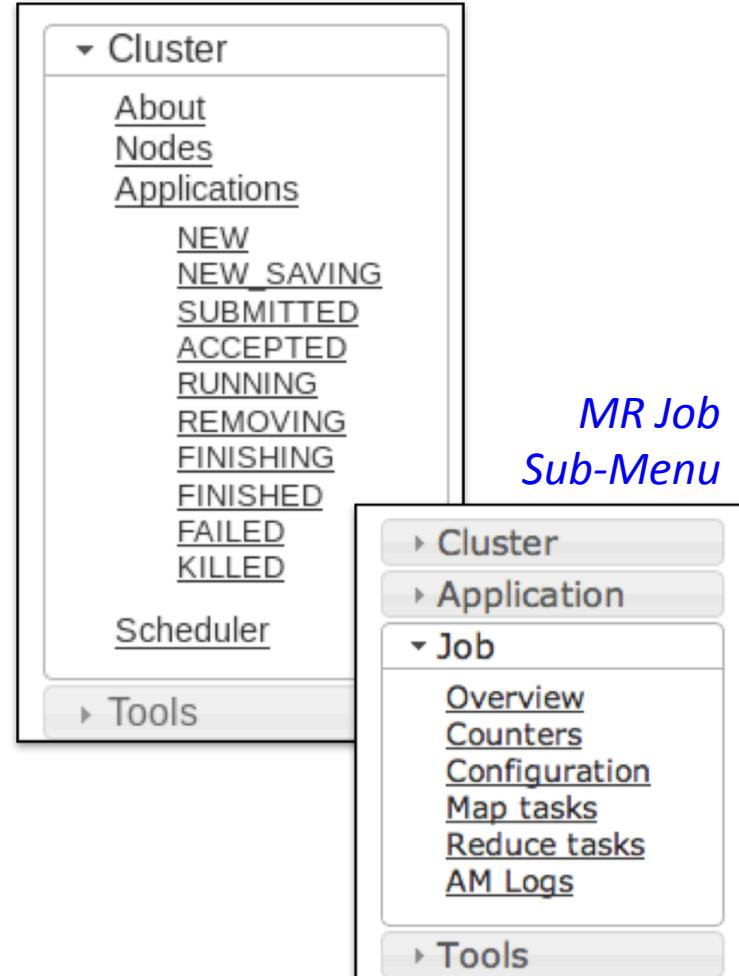
Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	saveAsTextFile at <console>:17	2015/01/13 21:54:01	22 ms	2/2 (1 skipped)	4/4 (2 skipped)

http://shs_host:18088

The ResourceManager Web UI

- The ResourceManager Web UI Menu
- Choose a link under “Applications” (e.g., “RUNNING” or “FINISHED”)
- Then click on the “application ID” to see application metrics including:
 - Links to app-specific logs
 - If the app has *completed*, an app “history” link that takes you to the applicable history server web UI
 - If the app is *still running*, the “Tracking URL” links to either...
 - The MR Job details in the same UI
 - The Spark Job details in a Spark shell application UI

Main ResourceManager
Web UI Menu



MapReduce Job HistoryServer Web UI

- The ResourceManager does not keep track of job history
- HistoryServer Web UI for MapReduce
 - Archives jobs' metrics and metadata
 - Can be accessed through Job History Web UI or Hue



<http://rmhost:19888/jobhistory>

The screenshot shows the Hadoop JobHistory Web UI interface. At the top, there's a navigation bar with the Hadoop logo, a search bar, and a user logged-in indicator. Below the header, a table displays "Retired Jobs". The table has columns for Start Time, Finish Time, Job ID, Name, User, Queue, State, Maps Total, Maps Completed, Reduces Total, and Reduces Completed. Five job entries are listed, each with a blue dashed border around it:

Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Completed	Reduces Total	Reduces Completed
2013.11.21 13:07:38 PST	2013.11.21 13:08:27 PST	job_1385066116114_0004	Process Logs	cloudera	default	SUCCEEDED	4	4	12	12
2013.11.21 13:03:53 PST	2013.11.21 13:04:42 PST	job_1385066116114_0003	Process Logs	cloudera	default	SUCCEEDED	4	4	12	12
2013.11.21 13:01:35 PST	2013.11.21 13:02:28 PST	job_1385066116114_0002	Process Logs	cloudera	default	SUCCEEDED	4	4	12	12
2013.11.21 12:48:00 PST	2013.11.21 12:50:43 PST	job_1385066116114_0001	Word Count	cloudera	default	SUCCEEDED	4	4	1	1
2013.11.21 09:24:45 PST	2013.11.21 09:28:19 PST	job_1385049040288_0003	Word Count	cloudera	default	SUCCEEDED	4	4	1	1

Spark History Server Web UI

The screenshot shows the Spark History Server Web UI interface. At the top, there is a navigation bar with a logo and the URL `http://sparkHistoryServerHost:18088`. Below the URL, the title "Spark Jobs" is displayed with a help icon. A summary section shows scheduling mode (FIFO), active jobs (0), completed jobs (2), and failed jobs (0). The "Completed Jobs" section lists two entries: "saveAsTextFile at <console>:17" and "sortByKey at <console>:16". The "Failed Jobs" section shows 0 failed jobs.

Spark shell (application_1421209... application)

Spark Jobs [\(?\)](#)

Scheduling Mode: FIFO

Active Jobs: 0

Completed Jobs: 2

Failed Jobs: 0

Active Jobs (0)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
--------	-------------	-----------	----------	-------------------------	---

Completed Jobs (2)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	saveAsTextFile at <console>:17	2015/01/13 21:54:01	22 ms	2/2 (1 skipped)	4/4 (2 skipped)
0	sortByKey at <console>:16	2015/01/13 21:54:01	81 ms	2/2	4/4

Failed Jobs (0)

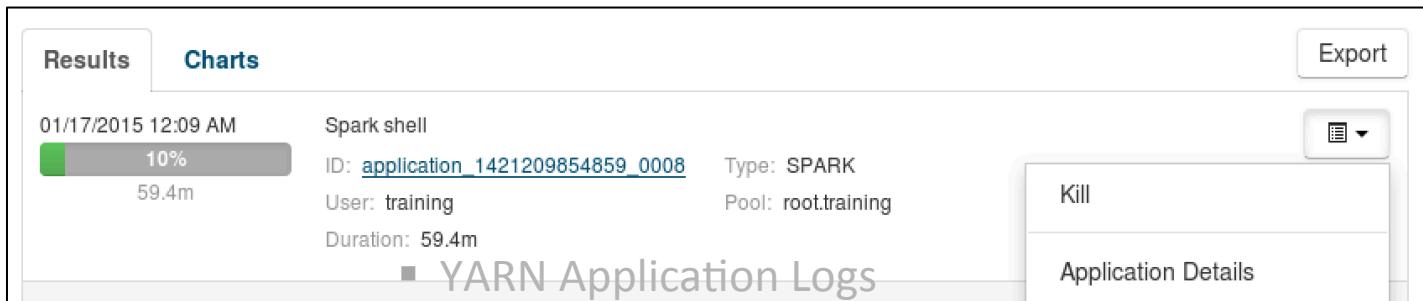
Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
--------	-------------	-----------	----------	-------------------------	---

Discovering YARN Application Details in the Shell

- **Displaying YARN application details in the shell**
 - To view all applications currently running on the cluster
 - `yarn application -list`
 - Returns all running applications, including the application ID for each
 - To view all applications on the cluster, including completed applications
 - `yarn application -list all`
 - To display the status of an individual application
 - `yarn application -status <application_ID>`

Killing YARN Applications

- To kill a running YARN application from Cloudera Manager
 - Use the drop down menu for the application in the YARN Applications page for the cluster



- To kill a running YARN application from the command line
 - You *can not* kill it just by hitting CTRL-C in the terminal
 - This only kills the client that launched the application
 - The application is still running on the cluster!
 - Utilize the YARN command line utility instead
 - `yarn application -kill <application_ID>`

Chapter Topics

MapReduce and Spark on YARN

- The Role of Computational Frameworks
- YARN: The Cluster Resource Manager
- MapReduce Concepts
- Apache Spark Concepts
- Running Computational Frameworks on YARN
- Exploring YARN Applications Through the Web UIs, and the Shell
- **YARN Application Logs**
- Conclusion
- Hands-On Exercise: Running YARN Applications

YARN Application Log Aggregation

- **Debugging distributed processes is intrinsically difficult**
 - Cloudera Manager's log aggregation improves this situation
- **YARN provides application log aggregation services**
 - Recommended (and enabled by default by Cloudera Manager)
 - Logs are aggregated by application
 - Access the logs from Cloudera Manager, any HDFS client, or the shell
- **When YARN log aggregation is enabled:**
 - Container log files are moved from NodeManager hosts' /var/log/hadoop-yarn/container directories to HDFS when the application completes
 - Default HDFS directory: /tmp/logs
- **Aggregated YARN application log retention**
 - Apache default: log retention disabled (and infinite when enabled)
 - Cloudera Manager default: enabled and 7 day retention

Accessing YARN Application Logs in Cloudera Manager

- Access all application logs in Cloudera Manager
 - From the YARN Applications page, choose “Collect diagnostics data”
 - Options to “Download Result Data”, and view recent or full logs

The screenshot shows two pages from Cloudera Manager:

- YARN Applications Page:** Shows a list of applications. One application is selected, and a context menu is open over it. The "Collect diagnostics data" option in the menu is highlighted with a red box.
- Command Details: YARN application diagnostics collection Page:** Displays the status of the diagnostics collection command. It includes:
 - Command:** YARN application diagnostics collection.
 - Context:** YARN (MR2 Included)
 - Status:** Finished
 - Started at:** Feb 10, 2015 2:18:46 AM PST
 - Ended at:** Feb 10, 2015 2:19:01 AM PSTA red arrow points from the "Collect diagnostics data" menu option to this page.

Both pages have a red box highlighting the "Download Result Data" button at the bottom.

MapReduce Log Details

- **MapReduce jobs produce the following logs:**
 - ApplicationMaster log
 - stdout, stderr, and syslog output for each Map and Reduce task
 - Job configuration settings specified by the developer
 - Counters
- **MapReduce Job History Server log directory**
 - Default: /var/log/hadoop-mapreduce

Spark Log Details

- **Spark can write application history logs to HDFS**
 - Default: enabled
- **Spark application history location**
 - Default location in HDFS: /user/spark/applicationHistory
 - Find and view the logs from the command line:
 - \$ sudo -u hdfs hdfs dfs /user/spark/\\applicationHistory
 - \$ sudo -u hdfs hdfs dfs -cat /user/spark/\\application_<application_id>/EVENT_LOG_1
- **Spark History Server log directory**
 - Default: /var/log/spark

Accessing YARN Application Logs From the Command Line

- Accessing application logs from the command line allows you to use utilities like grep for quick log analysis
- First find the application ID with `yarn application -list`

```
$ yarn application -list -appStates FINISHED | grep 'word count'  
application_1392918622651_0004 word count MAPREDUCE  
          training      root.training FINISHED SUCCEEDED  
100% http://monkey:19888/jobhistory/job/job_1392918622651_0004
```

- Then, once you have the application ID, you can list its logs

```
$ yarn logs -applicationId application_1392918622651_0004  
...  
Container: container_1392918622651_0004_01_000002 on elephant_46591  
=====  
...  
LogType: syslog  
LogLength: 4085  
Log Contents:  
2014-02-20 16:59:20,462 WARN [main] org.apache.hadoop.conf.Configuration:  
job.xml:an attempt to override final parameter: mapreduce.job.end-  
notification.max.retry.interval; Ignoring.
```

Chapter Topics

MapReduce and Spark on YARN

- The Role of Computational Frameworks
- YARN: The Cluster Resource Manager
- MapReduce Concepts
- Apache Spark Concepts
- Running Computational Frameworks on YARN
- Exploring YARN Applications Through the Web UIs, and the Shell
- YARN Application Logs
- **Conclusion**
- Hands-On Exercise: Running YARN Applications

Essential Points (1)

- **Resources on the cluster are managed by YARN:**
 - YARN ResourceManager schedules resources and manages the application lifecycle
 - YARN NodeManagers launch containers
- **Computational frameworks such as MapReduce and Spark use YARN**
 - Developers don't need to deal with cluster resource management, scheduling, fault tolerance, etc.
- **Individual tasks in a YARN application are managed by an ApplicationMaster**
- **A MapReduce job has a map phase, a shuffle and sort phase, and a Reduce phase**

Essential Points (2)

- **A Spark application consists of one or more jobs that run one or more stages of tasks**
- **There are many UIs available to observe details about YARN applications**
 - Cloudera Manager's YARN Applications page
 - Resource Manager Web UI
 - The (MapReduce) Job History Server Web UI
 - The Spark History Server Web UI
- **YARN application logs are aggregated by Cloudera Manager by default and are accessible from Cloudera Manager, HDFS clients, and the command line**

Chapter Topics

MapReduce and Spark on YARN

- The Role of Computational Frameworks
- YARN: The Cluster Resource Manager
- MapReduce Concepts
- Apache Spark Concepts
- Running Computational Frameworks on YARN
- Other Features of YARN
- Exploring YARN Applications Through the Web UIs, and the Shell
- Conclusion
- **Hands-On Exercise: Running YARN Applications**

Hands-On Exercise: Running YARN Applications

- In this Hands-On Exercise, you will run a MapReduce Job and examine the results in HDFS and in the logs. You will then install the Spark service and run a Spark on YARN application and observe the results
- Please refer to the Hands-On Exercise Manual



Hadoop Configuration and Daemon Logs

Chapter 6



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- **Hadoop Configuration and Daemon Logs**
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Hadoop Configuration and Daemon Logs

In this chapter, you will learn:

- **How to manage Hadoop cluster configurations using Cloudera Manager**
- **How to locate a configuration setting and deploy setting changes**
- **How to manage role instances and add services to a cluster**
- **Recommended HDFS service settings**
- **Recommended YARN service settings**
- **How to locate and access Hadoop daemon logs**

Chapter Topics

Hadoop Configuration and Logging

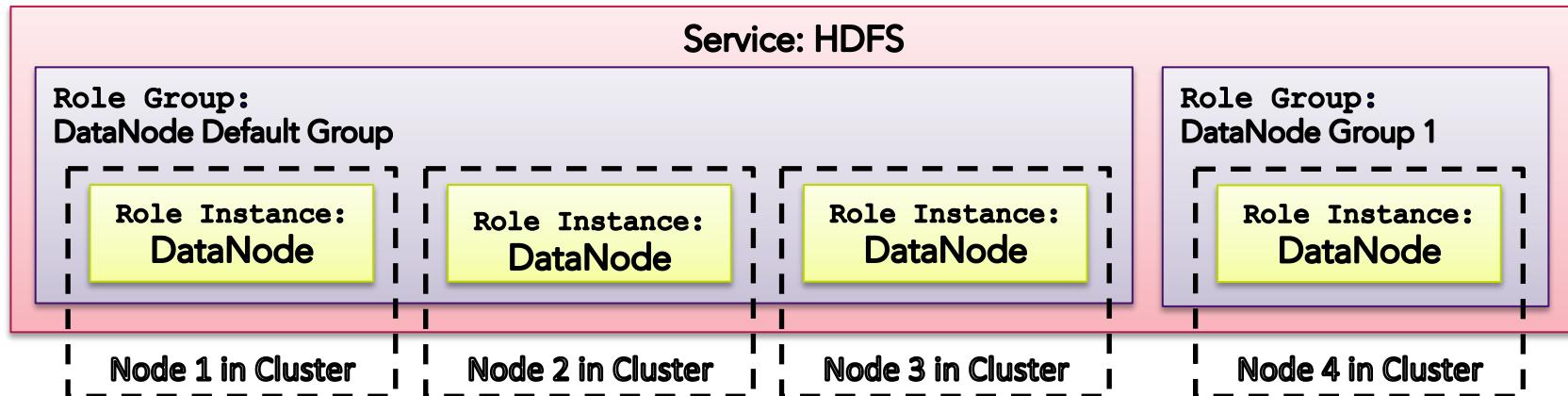
- **Cloudera Manager Constructs for Managing Configurations**
- Locating Configurations and Applying Configuration Changes
- Managing Role Instances and Adding Services
- Configuring the HDFS Service
- Configuring Hadoop Daemon Logs
- Configuring the YARN Service
- Conclusion
- Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

Configuring Hadoop Clusters

- Once Hadoop services are installed on a cluster, you will probably want to configure changes over time
- Common examples:
 - Change the settings of an existing Hadoop service
 - Add Hadoop services to a cluster
 - Add new DataNodes to a cluster
 - Remove nodes from a cluster
 - Upgrade software versions installed on nodes in the cluster
- Cloudera Manager makes configuration changes relatively simple to implement and track
 - Constructs such as “Role Groups” assist with cluster management
 - Group machines with similar configurations

Cloudera Manager – Configuration Terminology

- **Services** – A Hadoop cluster has Services, such as HDFS or YARN
- **Roles** – created by Cloudera Manager when a service is configured. For example, NameNode is a *role* in the HDFS service
- **Role Group** – a Cloudera Manager construct for managing configurations across multiple machines. Each role group has its own set of configurations
- **Role Instance** – a member of a Role Group. For example, a DataNode installed on a specific machine is a role instance



Role Groups

- **Role groups allow grouping roles of the same type that need to be configured in the same way**
- **Eases configuration management**
 - Group roles running on hosts with similar hardware configurations
 - Group roles co-located with the same set of other roles
- Create new role groups as needed
 - Cloudera Manager automatically creates role groups when services are created
 - A role instance can be moved to a different role group

Cloudera Manager Configuration menus are organized by role groups [example HDFS Configuration Menu]

Category
► Service-Wide
► Balancer Default Group
▼ DataNode Default Group
Advanced
Logs
Monitoring
Performance
Plugins
Ports and Addresses
Resource Management
Security
Stacks Collection
► DataNode Group 1

Configuration Settings – Inheritance

- **Configuration settings: order of inheritance**
 - Service -> Role Group -> Role Instance
- **A setting at the Role Group level will override the Service level setting**
 - Example: assume some DataNode hosts in the cluster have more HDFS storage capacity than other DataNode hosts in the cluster
 - Configure a DataNode Role Group for the larger hosts with storage settings that override the lower default service-level storage capacity settings
 - Add the larger hosts' DataNode roles to the Role Group
- **A setting at the Role Instance level will override the Role Group level and Service level settings**
 - Example: temporarily enable debug logging for a role instance while troubleshooting a potential hardware problem

Default Settings

- Cloudera Manager auto-configures role groups during installation, based on the resources available
- Every non-default setting has a useful “Reset to the default” option
- Easy to find all default setting overrides in the cluster
 - From the Cluster page, choose “Configuration > All Non-default Values”

The screenshot shows the Cloudera Manager web interface. At the top, there's a navigation bar with links for Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. Below the navigation bar, the main area is titled "Cluster 1 Status". A dropdown menu is open under the "Configuration" tab, with the option "All Non-default Values" highlighted by a red box. Other options in the dropdown include "All Log Directories", "All Disk Space Thresholds", and "All Port Configurations". To the right of the dropdown, there's a "Save Changes" button. On the left side of the main area, there are sections for "Configuration" (with "Filters" and search/status dropdowns), "DataNode Data Directory" (set to "dfs.data.dir, dfs.datanode.data.dir" with a value input field containing "/dfs/dn" and +/- buttons), and "DataNode Default Group" (set to "DataNode Default Group").

Cluster-Wide Configurations

- Access from Cloudera Manager's Cluster page Configuration menu
 - Review and/or modify...
 - All non-default value configurations on the cluster
 - All log directories
 - All disk space thresholds
 - All port configurations

The screenshot shows the Cloudera Manager interface for 'Cluster 1'. The top navigation bar includes links for Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. The 'Clusters' link is currently selected. On the left, there's a 'Status' section with icons for Hosts, HDFS, Spark, and YARN (MR2 Incl...). A dropdown menu is open over the 'Configuration' link, listing four options: 'All Non-default Values', 'All Log Directories', 'All Disk Space Thresholds', and 'All Port Configurations'. To the right, there's a 'Charts' section titled 'Cluster CPU' with a chart area showing performance metrics over time intervals of 30m, 1h, 2h, 6h, 12h, 1d, 7d, and 30d.

Chapter Topics

Hadoop Configuration and Logging

- Cloudera Manager Constructs for Managing Configurations
- **Locating Configurations and Applying Configuration Changes**
- Managing Role Instances and Adding Services
- Configuring the HDFS Service
- Configuring Hadoop Daemon Logs
- Configuring the YARN Service
- Conclusion
- Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

Using Global Search to Locate a Configuration Property

The screenshot shows the Cloudera Manager interface for Cluster 1. The left sidebar has 'HDFS Status' selected under 'Configuration'. A search bar at the top contains the query 'replication fac'. Below it, a list of configuration properties is displayed, with several items highlighted in blue, indicating they are being searched for. The properties listed include various replication factors and thresholds across different HDFS components like Replication Sources, Schedules, and Minimal/Maximal Block Replication.

Category	Property	Value	Description
global	Replication Sources		
global	Replication Schedules		
config	HDFS: Replication Factor		
config	YARN (MR2 Included): Mapreduce Submit Replication		
config	HDFS: Minimal Block Replication		
config	HDFS: Maximal Block Replication		
config	HDFS: HDFS Replication Advanced Configuration Snippet (Safety Valve)		
config	Cloudera Management Service: Index Merge Factor		
config	HDFS: Replication Work Multiplier Per Iteration		
config	HDFS: Safemode Threshold Percentage		
config	YARN (MR2 Included): I/O Sort Factor		
config	Cloudera Management Service: Cgroup Memory Soft Limit		
config	Cloudera Management Service: Cgroup Memory Soft Limit		
config	Cloudera Management Service: Cgroup Memory Soft Limit		
chart	HDFS Pending Replication Blocks		
chart	HDFS Scheduled Replication Blocks		
	Default Umask dfs.umaskmode, fs.permissions.umask-mode	022 default value	Default umask for file and directory creation, specified in an octal value (with a leading 0)
	Enable WebHDFS dfs.webhdfs.enabled	<input checked="" type="checkbox"/> default value	Enable WebHDFS interface

- Use Cloudera Manager global search to locate a configuration property
 - Or browse to <cluster's service> Configuration tab to find settings

Making a Configuration Change

The screenshot shows the Cloudera Manager interface for the HDFS service. The 'Configuration' tab is selected. A validation warning is present. The 'Replication Factor' property is being edited, with its value set to 4. The 'Save Changes' button is highlighted with a red box.

Category	Property	Value	Description
Service-Wide	Replication Factor dfs.replication	4 Reset to the default value: 3 ↵	Default block replication. The number of replications to make when the file is created. The default value is used if a replication number is not specified.
	Minimal Block Replication dfs.replication.min, dfs.namenode.replication.min	1 default value	The minimal block replication.
	Maximal Block Replication dfs.replication.max	512 default value	The maximal block replication.

Stale Configurations

- When you modify a configuration and click “Save changes”, it may require client redeployment and/or a role instance restart or refresh
- The affected service(s) will display one or more of the icons below
- Click the icon to display the Stale Configurations page



Stale configuration – Restart Required



Stale configuration – Refresh Required



Stale configuration – Client configuration redeployment needed

Applying a Configuration Change

- Cloudera Manager’s “Stale Configurations – Review Changes” screen
 - Shows which changes will be applied to which role instance(s)
 - Prompts you to “Restart Cluster” if necessary

The screenshot shows the Cloudera Manager interface for reviewing stale configurations. The top navigation bar includes Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. The main content area is titled "Cluster 1 Stale Configurations". It displays two configuration files:

- File: hadoop-conf/hdfs-site.xml**

```
...  ... @@ -27,9 +27,9 @@
27 27      <value>3</value>
28 28    </property>
29 29    <property>
30 30      <name>dfs.blocksize</name>
31 31      <value>134217728</value>
31 31      + <value>268435456</value>
32 32    </property>
33 33    <property>
34 34      <name>dfs.client.use.datanode.hostname</name>
35 35      <value>false</value>
```
- File: hdfs-site.xml**

```
...  ... @@ -31,9 +31,9 @@
31 31      <value>512</value>
32 32    </property>
33 33    <property>
34 34      <name>dfs.blocksize</name>
35 35      <value>134217728</value>
35 35      + <value>268435456</value>
```

A red box highlights the "Restart Cluster" button in the top right corner of the review panel.

Service Configuration Settings – Deployment

- **Key point: each daemon configuration is given its own execution and configuration environment**
 - This is very different than in non-Cloudera Manager managed clusters
- **Configuration files for daemons that Cloudera Manager manages are deployed on any host that has a role for the service**
- **The Cloudera Manager Agent pulls the daemon configuration and places it in the appropriate configuration directory**
 - Includes arguments to exec(), config files, directories to create, and other information (such as cgroup settings)
- **Default location for Hadoop daemon configurations**
 - A subdirectory of
`/var/run/cloudera-scm-agent/process/`

Client Configuration Files – Deployment

- **Cloudera Manager stores client configurations separately from service configurations**
 - Default location: /etc/hadoop/conf
- **Cloudera Manager creates a “client configuration file” zip archive of the configuration files containing service properties**
 - Each archive has the configuration files needed to access the service
 - Example: a MapReduce client configuration file contains copies of core-site.xml, hadoop-env.sh, hdfs-site.xml, log4j.properties and mapred-site.xml
- **Client configuration files are generated by Cloudera Manager when a cluster is created, or a service or a gateway role is added on a host**
 - A “gateway” role has client configurations, but no Hadoop daemons
 - Cloudera Manager indicates when Client Configuration Redeployment is needed. There is also an option to download and manually deploy

Summary: Server and Client Settings are Maintained Separately

- **Cloudera Manager decouples server and client configurations**

- Server settings (e.g., NameNode, DataNode) are stored in /var/run/cloudera-scm-agent/process subdirectories
- Client settings are stored in /etc/hadoop subdirectories

```
training@tiger:/etc/hadoop$ tree conf
conf
├── _cloudera_generation_
├── container-executor.cfg
├── core-site.xml
├── hadoop-env.sh
├── hdfs-site.xml
├── log4j.properties
├── mapred-site.xml
├── ssl-client.xml
├── topology.map
├── topology.py
└── yarn-site.xml
```

```
training@tiger:/var/run/cloudera-scm-agent/process$ sudo tree
.
├── 10-hdfs-DATANODE
│   ├── cloudera-monitor.properties
│   ├── cloudera-stack-monitor.properties
│   ├── core-site.xml
│   ├── event-filter-rules.json
│   ├── hadoop-metrics2.properties
│   ├── hadoop-policy.xml
│   ├── hdfs.keytab
│   ├── hdfs-site.xml
│   ├── http-auth-signature-secret
│   ├── log4j.properties
│   ├── logs
│   │   ├── stderr.log
│   │   └── stdout.log
│   ├── ssl-client.xml
│   └── ssl-server.xml
└── 11-hdfs-SECONDARYNAMENODE
    ├── cloudera-monitor.properties
    ├── cloudera-stack-monitor.properties
    ├── core-site.xml
    ├── event-filter-rules.json
    ├── hadoop-metrics2.properties
    ├── hadoop-policy.xml
    ├── hdfs.keytab
    ├── hdfs-site.xml
    ├── http-auth-signature-secret
    ├── log4j.properties
    ├── logs
    │   ├── stderr.log
    │   └── stdout.log
    ├── ssl-client.xml
    └── ssl-server.xml
```

Chapter Topics

Hadoop Configuration and Logging

- Cloudera Manager Constructs for Managing Configurations
- Locating Configurations and Applying Configuration Changes
- **Managing Role Instances and Adding Services**
- Configuring the HDFS Service
- Configuring Hadoop Daemon Logs
- Configuring the YARN Service
- Conclusion
- Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

Managing Role Instances

- **Add a role instance**
 - After a service is created, add more role instances to the cluster
 - Example: add a DataNode role instance to a host
- **Start, Stop, Restart role instances from Cloudera Manager**
- **Delete a role instance**
 - Host will no longer be managed by Cloudera Manager
 - Does not delete the deployed client configurations
- **Decommission a role instance**
 - Remove role instance (or even an entire cluster host) from a cluster while it is running without losing data
 - Decommissioning a DataNode is not possible if doing so would drop the number of remaining DataNodes below the replication factor

Adding Services to a Cluster

- **Use the ‘Add a Service’ wizard**

- Choose ‘Add a Service’
- Confirm dependencies
- Choose host(s) to run the service
- Review configurations
- The service is deployed on node(s)

cloudera manager

Add Service Wizard

Select the type of service you want to add.

Service Type	Description
<input type="radio"/>  Accumulo 1.6	The Apache Accumulo storage and retrieval system.
<input type="radio"/>  Flume	Flume collects and moves large volumes of event data.
<input type="radio"/>  HBase	Apache HBase provides distributed, column-oriented storage designed to handle very large amounts of structured data in real time (via ZooKeeper).
<input type="radio"/>  HDFS	Apache Hadoop Distributed File System (HDFS) stores data across multiple hosts and provides reliable storage for applications. HDFS is designed to scale to thousands of hosts and petabytes of data throughout a cluster.

- **Add-on services**

- Non-CDH software managed with Cloudera Manager
- Provided by an independent software vendor or by Cloudera
- Require a Custom Service Descriptor (CSD) JAR file

Chapter Topics

Hadoop Configuration and Logging

- Cloudera Manager Constructs for Managing Configurations
- Locating Configurations and Applying Configuration Changes
- Managing Role Instances and Adding Services
- **Configuring the HDFS Service**
- Configuring Hadoop Daemon Logs
- Configuring the YARN Service
- Conclusion
- Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

Default Settings and Cloudera Manager Initial Settings

- There are many Hadoop parameters that can be set
- Non-Cloudera Manager default CDH settings are documented at <http://archive.cloudera.com/cdh5/cdh/5/hadoop>
 - Click the links under Configuration
- Cloudera Manager *defines its own default CDH settings*
 - CM default values are often different than non-CM “Hadoop” defaults
 - CM defaults are based on Cloudera’s experience
- Values that CM automatically configures during installation are not always CM’s ‘default’ values
 - Some initial values are set by CM based on factors such as hardware and node count
- The “reset to default” links in Cloudera Manager resets parameters to a static value regardless of hardware and node count

Recommended Parameter Values

- Here we present some of the key parameters, and suggest recommended values
 - Based on our experiences working with clusters ranging from a few nodes up to 1,500+

HDFS NameNode Metadata Location (1)

- The single most important configuration value on your entire cluster, used by the NameNode:

dfs.namenode.name.dir

Set in HDFS / NameNode Group

- Location on the local filesystem where the NameNode stores its name table (fsimage) metadata
- Multiple entries in Cloudera Manager
- Default is an empty value

- Loss of a NameNode's metadata will result in the effective loss of all the data in its namespace
 - Although the blocks will remain, there is no way of reconstructing the original files without the metadata

HDFS NameNode Metadata Location (2)

- **For all HDFS deployments, `dfs.namenode.name.dir` must specify at least two disks (or a RAID volume) on the NameNode**
 - Failure to set correctly **will** result in eventual loss of your cluster's data
- **By default, a NameNode will write to the edit log files in all directories in `dfs.namenode.name.dir` synchronously**
 - For non-HA HDFS deployments, can explicitly specify the path to the edits log directory by setting `dfs.namenode.edits.dir`
 - High-Availability HDFS settings are discussed in a later chapter
- **If a directory in the list disappears, the NameNode will continue to function**
 - It will ignore that directory until it is restarted

HDFS NameNode Metadata Location (3)

- **For non-HA HDFS deployments, `dfs.namenode.name.dir` must additionally specify an NFS mount elsewhere on the network**
 - If you do not do this, catastrophic failure of the NameNode will result in the loss of the metadata
- **Recommendation for the NFS mount point**

`tcp,soft,intr,timeo=10,retrans=10`

 - Soft mount so the NameNode will not hang if the mount point disappears
 - Will retry transactions 10 times, at 1-10 second intervals, before being deemed to have failed

HDFS DataNode Configuration

dfs.datanode.du.reserved

Set in HDFS / DataNode Group

- The amount of space on each disk in `dfs.datanode.data.dir` which cannot be used for HDFS block storage
- Reserve space typically used for Mapper and Reducer intermediate data
- Default: 10GB
- Recommended: 25% of the space on each volume, with a minimum of 10GB

dfs.datanode.data.dir

Set in HDFS / DataNode Group

- Location on the local filesystem where a DataNode stores its blocks
- Can be a list of directories; round-robin writes to the directories (no redundancy)
- Can be different on each DataNode

HDFS Block Size and Replication Factor

dfs.blocksize

Set in HDFS / Service-Wide

- The block size for new files, in bytes
- Default and recommended value is 134217728 (128MB)
- Note: this is a client-side setting

dfs.replication

Set in HDFS / Service-Wide / Replication

- The number of times each block should be replicated when a file is written
- Default and recommended value is 3
- Note: this is a client-side setting

HDFS Trash Interval

Use Trash

Set in HDFS / Gateway Group

- When a file is deleted using the Hadoop shell, it is moved to the `.Trash` directory in the user's home instead of being immediately deleted
- Supports data recovery
- Default in Cloudera Manager: true
- Recommendation: true
- This is a client-side setting and is used by the NameNode
- To recover a file, move it to a location outside the `.Trash` subdirectory

`fs.trash.interval`

Set in HDFS / NameNode Group

- Specifies the minimum number of minutes files will be kept in the trash
- CM Default : 1 day (1440 minutes)

HDFS Memory Settings

- Ideal heap size will vary by use case
- Recommended to reserve 3GB of available memory for the OS
- NameNode – 1GB per million blocks (recommended minimum 4GB)
- DataNode – 1GB min, 4GB max

Service	Setting	Default
DataNode	Java Heap Size of DataNode	1GB
DataNode	Map Memory for Caching Data Blocks	4GB
Failover Controller*	Java Heap Size	256MB
JournalNode*	Java Heap Size	256MB
NameNode	Java Heap Size	1GB
HDFS Balancer	Java Heap Size of Balancer	1GB

* Applicable to HA HDFS only

Chapter Topics

Hadoop Configuration and Logging

- Cloudera Manager Constructs for Managing Configurations
- Locating Configurations and Applying Configuration Changes
- Managing Role Instances and Adding Services
- Configuring the HDFS Service
- **Configuring Hadoop Daemon Logs**
- Configuring the YARN Service
- Conclusion
- Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

Daemon Logs vs Application Logs

- **Hadoop Daemon logs**
 - Include the logs for the NameNode, DataNodes, NodeManager, JobHistory Server, etc.
- **Application logs**
 - Include logs from YARN containers
 - Application Masters, MapReduce tasks, Spark executors
 - Discussed in an earlier chapter
- **All logs have these threshold options:**
 - TRACE, DEBUG, INFO, WARN, ERROR, FATAL
 - INFO is the default

Log Management

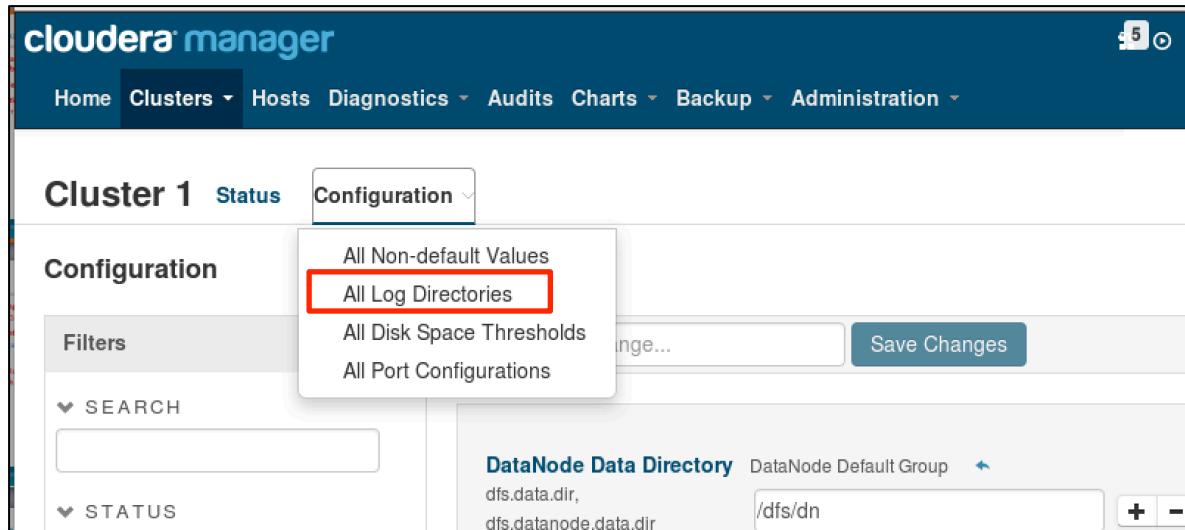
- **Every Hadoop cluster daemon writes to a different log file**
 - Without log management:
 - It can be difficult to know where to look for critical logged information about your cluster
 - Requires the administrator to ssh into individual machines to examine the log files
 - It is difficult to consolidate logs from the same time period across different machines
- **Cloudera Manager provides comprehensive log management**
 - Contextualizes all system logs from across the cluster
 - Allows you to search and filter by service, role, host, keyword, and severity

Daemon Logs – Retention

- **Configure log retention in the relevant Role Groups**
 - In YARN Configuration
 - NodeManager, ResourceManager, JobHistory Server
 - In HDFS Configuration
 - DataNode, NameNode, JournalNode
- **Cloudera Manager Default Values**
 - Maximum size of generated log files = 200MB
 - Number of rolled log files retained = 10
 - Maximum disk space for logs for each component
 - 200MB per log x 10 rolled = 2GB

Hadoop Daemon Logs

- **Daemon logs exist on all machines running at least one Hadoop daemon**
 - Find/modify all log directory locations using Cloudera Manager



- **Every daemon writes to two files**
 - `.out` file – combination of `stdout` and `stderr` during daemon startup
 - `.log` file – first port of call when diagnosing problems

Changing Daemon Log Levels Persistently

- Persistent log level for each daemon configured in Cloudera Manager
 - In the Configuration area for each service, search “logging threshold”
- Default level is **INFO**

The screenshot shows the Cloudera Manager interface for the YARN service. The top navigation bar includes Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. Below the navigation is a breadcrumb trail: Cluster 1 > YARN (MR2 Included). The main content area is titled "Configuration". A search bar at the top of the table is set to "logging threshold". The table has three columns: Category, Property, and Value. There are four rows of data:

Category	Property	Value
JobHistory Server Default Group / Logs	JobHistory Server Logging Threshold	INFO default value
NodeManager Default Group / Logs	NodeManager Logging Threshold	INFO default value
NodeManager Group 1 / Logs	NodeManager Logging Threshold	INFO default value
ResourceManager Default Group / Logs	ResourceManager Logging Threshold	INFO

Consolidated and Searchable Logging (1)

- Logs are consolidated and searchable using Cloudera Manager
 - Choose Diagnostics > Logs
 - Set filters described below, enter search term(s), and click “Search”

The screenshot shows the Cloudera Manager interface for managing logs. On the left, there's a sidebar with a 'Diagnostics' dropdown menu open, showing options like 'Events', 'Logs' (which is selected and highlighted with a red box), and 'Server Log'. On the right, the main 'Logs' page is displayed. At the top, there's a search bar containing 'job' and a time range selector with options for 30m, 1h, 2h, 6h, 12h, 1d, 7d, and 30d. Below the search bar, there are sections for 'Select Sources' (with 'Hosts' selected and a search input field 'Enter a host name'), 'Minimum Log Level' (set to 'INFO'), and a list of log sources. The sources listed include 'Cluster 1' (with HDFS, Spark, and YARN checked), 'Cloudera Management Service' (with Cloudera Management Service checked), and 'Cloudera Manager' (with Cloudera Manager checked). A red box highlights the 'Select Sources' section. Numbered circles 1 through 4 point to the time range selector, the search bar, the minimum log level setting, and the source selection area respectively.

cloudera manager

Home Clusters Hosts **Diagnostics** Audits

Home Status All Help Logs Server Log

Cluster 1 (CDH 5.3.0 Parcels)

cloudera manager

Home Clusters Hosts **Diagnostics** Audits Charts Backup Administration

1 hour preceding January 21 2015, 11:02 PM PST

Logs

1 30m 1h 2h 6h 12h 1d 7d 30d

job

2 Search Additional Settings

Select Sources Hosts Enter a host name Minimum Log Level INFO

3

4 All Sources

Cluster 1

HDFS

Spark

YARN (MR2 Included)

Cloudera Management Service

Cloudera Manager

Cloudera Manager Service

Cloudera Manager

Previous Next

Machines Searched: 5, Errors: 0, Search Time: 309 ms, [More Statistics](#)

Host	Log Level	Time	Source	Message
monkey	INFO	January 21 2015 10:03 PM	JobHistory	Starting scan to move intermediate done files View Log File
monkey	INFO	January 21 2015 10:06 PM	JobHistory	Starting scan to move intermediate done files

Consolidated and Searchable Logging (2)

- **Search results include all log entries that match the filters applied**
 - Every log entry presented includes a “View Log File” link which opens the “Log Details” page
- **Log Details page shows a portion of the full log**
 - “Download Full Log” option available

The screenshot shows the Cloudera Manager interface. At the top, there's a navigation bar with links for Home, Clusters, Hosts, Diagnostics (which is currently selected), Audits, Charts, Backup, and Administration. Below the navigation bar is a search bar and a support link. The main content area is titled "Log Details". It displays log information for a host named "monkey". The log file path is listed as "/var/log/hadoop-mapreduce/hadoop-cmf-yarn-JOBHISTORY-monkey.log.out". The log entries are timestamped from January 16, 2015, to January 20, 2015. A red box highlights the "Download Full Log" button at the bottom right of the log table. An arrow points from the text "Download Full Log" in the list above to this button.

Time	Level	Category	Message
2:23:49.646 AM	ERROR	org.apache.hadoop.mapreduce.v2.hs.JobHistoryServer	RECEIVED SIGNAL 15: SIGTERM
2:23:49.672 AM	INFO	org.apache.hadoop.metrics2.impl.MetricsSystemImpl	Stopping JobHistoryServer metrics system.
2:23:49.675 AM	INFO	org.apache.hadoop.metrics2.impl.MetricsSystemImpl	JobHistoryServer metrics system stopped.
2:23:49.677 AM	INFO	org.apache.hadoop.metrics2.impl.MetricsSystemImpl	JobHistoryServer metrics system shutdown
2:23:49.688 AM	INFO	org.apache.hadoop.ipc.Server	Stopping server on 10033
2:23:49.691 AM	INFO	org.apache.hadoop.ipc.Server	Stopping IPC Server Responder
2:23:49.691 AM	INFO	org.apache.hadoop.ipc.Server	Stopping server on 10020
2:23:49.691 AM	INFO	org.apache.hadoop.ipc.Server	Stopping IPC Server listener on 10033
2:23:49.707 AM	INFO	org.apache.hadoop.ipc.Server	Stopping IPC Server Responder
2:23:49.710 AM	INFO	org.apache.hadoop.ipc.Server	Stopping IPC Server listener on 10020
2:23:49.718 AM	INFO	org.mortbay.log	Stopped HttpServer2\$SelectChannelConnector
2:23:49.723 AM	INFO	org.apache.hadoop.mapreduce.v2.hs.JobHistory	Stopping JobHistory
2:23:49.723 AM	INFO	org.apache.hadoop.mapreduce.v2.hs.JobHistory	Stopping History Cleaner/Move To Done

Chapter Topics

Hadoop Configuration and Logging

- Cloudera Manager Constructs for Managing Configurations
- Locating Configurations and Applying Configuration Changes
- Managing Role Instances and Adding Services
- Configuring the HDFS Service
- Configuring Hadoop Daemon Logs
- **Configuring the YARN Service**
- Conclusion
- Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

YARN Application Logging

- YARN generates logs for all the applications it runs
- Assists application developers and system administrators
- One log generated per YARN container
 - Log aggregation collects these together

YARN Application Logging Settings

yarn.log-aggregation-enable

Set in YARN / Service-Wide

- Enable log aggregation. Default: true. Recommendation: true.
- Used by the NodeManagers

yarn.nodemanager.remote-app-log-dir

Set in YARN / NodeManager Group

- HDFS directory to which application log files are aggregated
- Default: /tmp/logs. Example: /var/log/hadoop-yarn/apps
- Used by NodeManagers and the JobHistoryServer

yarn.nodemanager.log-dirs

Set in YARN / NodeManager Group

- Local directories to which application log files are written
- Note that with log aggregation enabled, these files are deleted after an application has completed
- Default: /var/log/hadoop-yarn/container
- Used by NodeManagers

YARN Resource Localization

yarn.nodemanager.local-dirs

Set in YARN / NodeManager Group

- Directories in which local resource files are stored
- In MapReduce, intermediate data files generated by the Mappers and used by the Reducers are local resource files
- Spark applications also make use of local resource files
- If these directories are not big enough, then jobs (applications) will fail
- Example: /disk1/yarn/nm, /disk2/yarn/nm
- Used by NodeManagers

Chapter Topics

Hadoop Configuration and Logging

- Cloudera Manager Constructs for Managing Configurations
- Locating Configurations and Applying Configuration Changes
- Managing Role Instances and Adding Services
- Configuring the HDFS Service
- Configuring Hadoop Daemon Logs
- Configuring the YARN Service
- **Conclusion**
- Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

Essential Points

- **Manage clusters and services in Cloudera Manager**
- **Role Groups and settings inheritance allow for organized management of configuration settings**
- **Hadoop configurations are configurable in Cloudera Manager**
 - Pushed to `/var/run/cloudera-scm-agent/process` sub-directory of cluster nodes where daemons run
- **Hadoop daemons log thresholds, locations, and retention are configurable**
- **Access daemon logs from Cloudera Manager, the NameNode Web UI, and the ResourceManager Web UI**

Chapter Topics

Hadoop Configuration and Logging

- Cloudera Manager Constructs for Managing Configurations
- Locating Configurations and Applying Configuration Changes
- Managing Role Instances and Adding Services
- Configuring the HDFS Service
- Configuring Hadoop Daemon Logs
- Configuring the YARN (MR2 Included) Service
- Conclusion
- **Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs**

Hands-On Exercise: Explore Hadoop Configurations and Logs

- In this Hands-On Exercise, you will apply a configuration change, observe where configuration settings are stored on disk, and examine Hadoop daemon log files
- Please refer to the Hands-On Exercise Manual



Getting Data Into HDFS

Chapter 7



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- **Getting Data Into HDFS**
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Getting Data Into HDFS

In this chapter, you will learn

- **How to import data into HDFS with Flume**
- **How to import data into HDFS with Sqoop**
- **What REST interfaces Hadoop provides**
- **Best practices for importing data**

Getting Data Into HDFS

- In an earlier chapter, you learned how to use the `hdfs dfs` command to copy data into and out of HDFS
- In this chapter, we will explore several Hadoop ecosystem tools for accessing HDFS data from outside of Hadoop
 - We can only provide a brief overview of these tools; consult the documentation at <http://docs.cloudera.com/> for full details on installation and configuration

Chapter Topics

Getting Data Into HDFS

- **Ingesting Data From External Sources With Flume**
- Hands-On Exercise: Using Flume
- Ingesting Data From Relational Databases With Sqoop
- REST Interfaces
- Best Practices for Importing Data
- Conclusion
- Hands-On Exercise: Importing Data With Sqoop

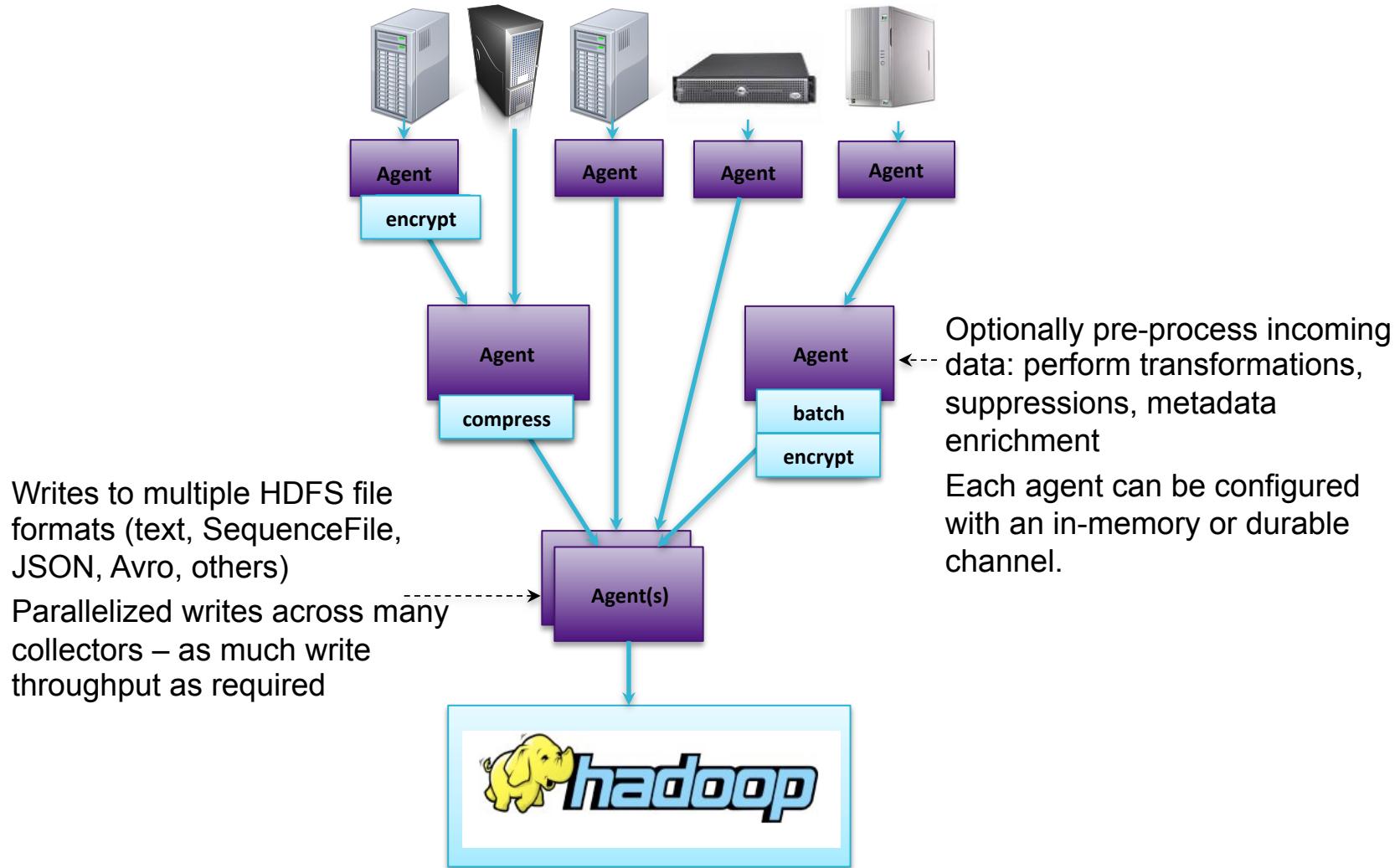
Apache Flume: Basics

- **Flume is a distributed, reliable, available service for efficiently moving large amounts of data as it is produced**
 - Ideally suited to gathering logs from multiple systems and inserting them into HDFS as they are generated
- **Flume is an open source Apache project**
 - Initially developed by Cloudera
 - Included in CDH
- **Flume's design goals:**
 - Reliability
 - Scalability
 - Extensibility

Flume: Usage Patterns

- Flume is typically used to ingest log files from real-time systems such as Web servers, firewalls, and sensor networks into HDFS
- Currently in use in many large organizations, ingesting millions of events per day

Flume: High-Level Overview



Flume Agent Characteristics

- Each Flume agent has a *source* and a *sink*
- Source
 - Tells the node where to receive data from
- Sink
 - Tells the node where to send data to
- Channel
 - A queue between the Source and Sink
 - Can be in-memory only or ‘Durable’
 - Durable channels will not lose data if power is lost



Flume's Design Goals: Reliability

- **Channels provide Flume's reliability**
- **Memory Channel**
 - Data will be lost if power is lost
- **Disk-based Channel**
 - Disk-based queue guarantees durability of data in case of a power loss
- **Data transfer between Agents and Channels is transactional**
 - A failed data transfer to a downstream agent rolls back and retries
- **Can configure multiple Agents with the same task**
 - e.g., two Agents doing the job of one 'collector' – if one agent fails then upstream agents would fail over

Flume's Design Goals: Scalability

- **Scalability**

- The ability to increase system performance linearly – or better – by adding more resources to the system
- Flume scales horizontally
 - As load increases, more machines can be added to the configuration

Flume's Design Goals: Extensibility

- **Extensibility**
 - The ability to add new functionality to a system
- **Flume can be extended by adding Sources and Sinks to existing storage layers or data platforms**
 - General Sources include data from files, syslog, and standard output from any Linux process
 - General Sinks include files on the local filesystem or HDFS
 - Developers can write their own Sources or Sinks

Installing, Configuring, and Starting Flume

- **Install (part of CDH)**
 - Use Cloudera Manager's 'Add a Service' wizard
 - If not using Cloudera Manager, install using package
- **Configure Agent Nodes in Cloudera Manager**
 - Go to Flume > Instances > Agent Configuration page
 - Agent source, sinks, channels, and flow settings deployed to
`/var/run/cloudera-scm-agent/process/nn-flume-AGENT`
- **Start Flume agent(s) in Cloudera Manager**

Agent Configuration Example (1)

- **tail -F of a file as the source, downstream node as the sink**

```
tail1.sources = src1
tail1.channels = ch1
tail1.sinks = sink1 sink2
tail1.sinkgroups = sg1

tail1.sources.src1.type = exec
tail1.sources.src1.command = tail -F /tmp/access_log
tail1.sources.src1.channels = ch1

tail1.channels.ch1.type = memory
tail1.channels.ch1.capacity = 500

tail1.sinks.sink1.type = avro
tail1.sinks.sink1.hostname = localhost
tail1.sinks.sink1.port = 6000
tail1.sinks.sink1.batch-size = 1
tail1.sinks.sink1.channel = ch1
```

Agent Configuration Example (2)

- Upstream node as the source, HDFS as the sink

```
collector1.sources = src1
collector1.channels = ch1
collector1.sinks = sink1

collector1.sources.src1.type = avro
collector1.sources.src1.bind = localhost
collector1.sources.src1.port = 6000
collector1.sources.src1.channels = ch1

collector1.channels.ch1.type = memory
collector1.channels.ch1.capacity = 500

collector1.sinks.sink1.type = hdfs
collector1.sinks.sink1.hdfs.path = /collector_dir
collector1.sinks.sink1.hdfs.filePrefix = access_log
collector1.sinks.sink1.channel = ch1
```

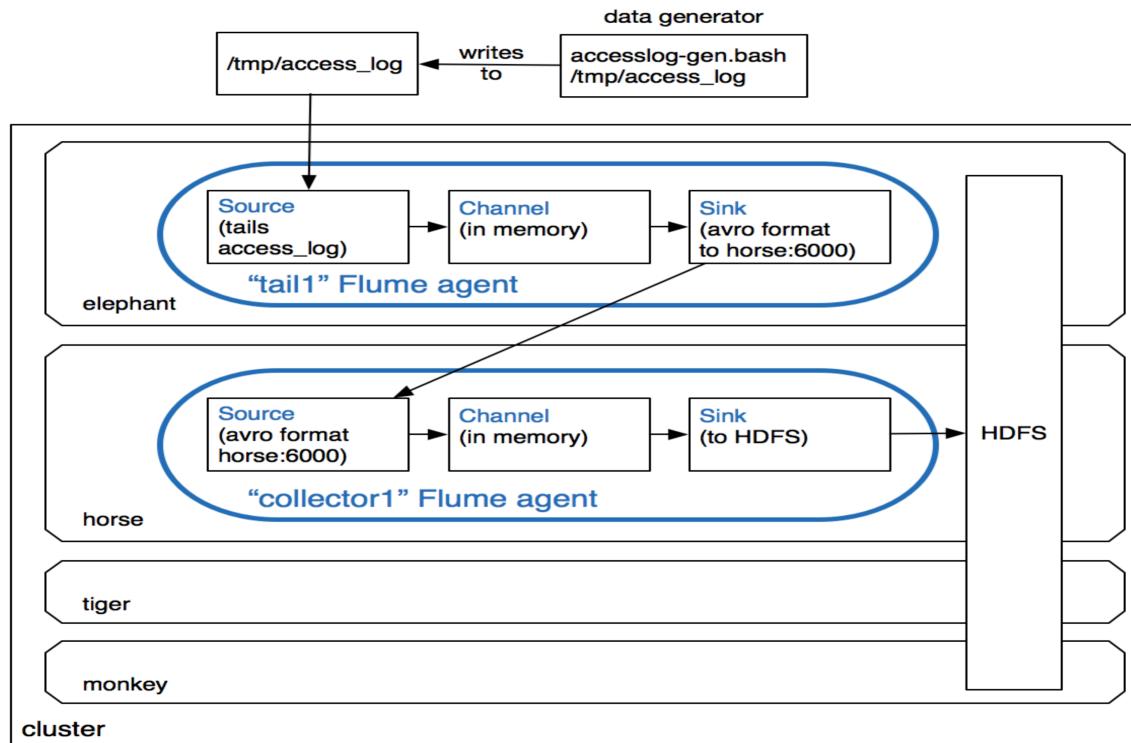
Chapter Topics

Getting Data Into HDFS

- Ingesting Data From External Sources With Flume
- **Hands-On Exercise: Using Flume**
- Ingesting Data From Relational Databases With Sqoop
- REST Interfaces
- Best Practices for Importing Data
- Conclusion
- Hands-On Exercise: Importing Data With Sqoop

Hands-On Exercise: Using Flume to Put Data into HDFS

- In this Hands-On Exercise, you will install the Flume service and then create two simple Flume agents configured to store dynamically-generated data in HDFS
- Please refer to the Hands-On Exercise Manual



Chapter Topics

Getting Data Into HDFS

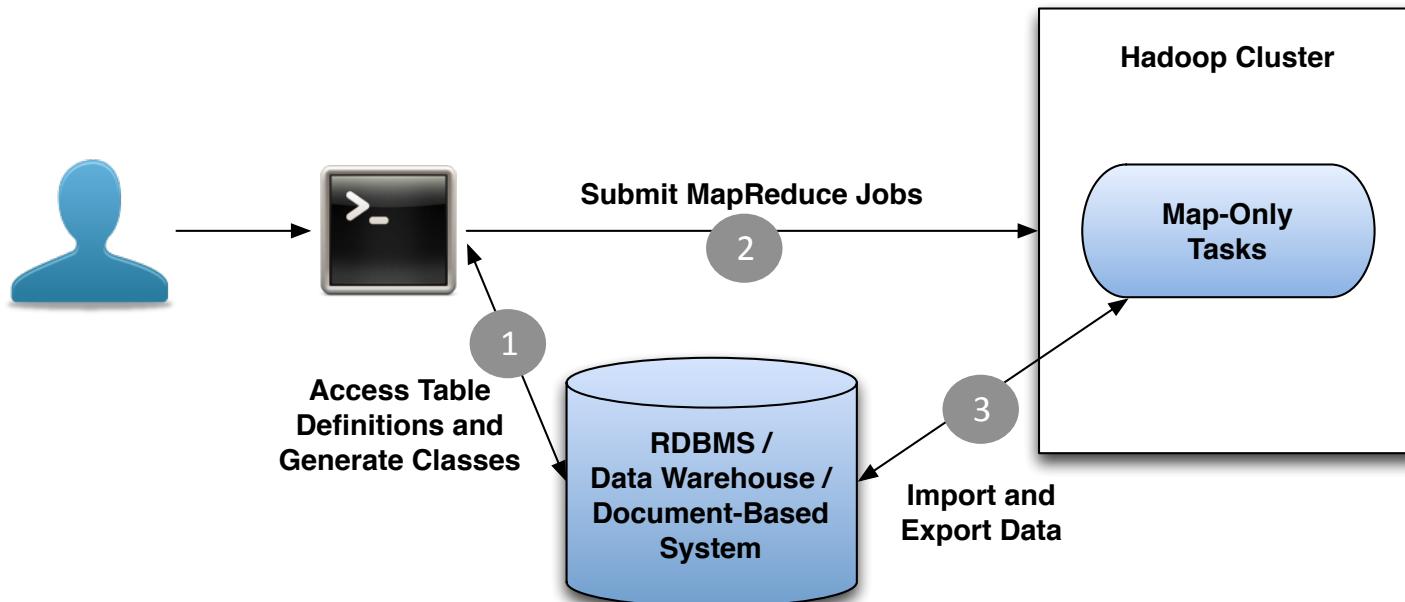
- Ingesting Data From External Sources With Flume
- Hands-On Exercise: Using Flume
- **Ingesting Data From Relational Databases With Sqoop**
- REST Interfaces
- Best Practices for Importing Data
- Conclusion
- Hands-On Exercise: Importing Data With Sqoop

What is Apache Sqoop?

- **Sqoop is “the SQL-to-Hadoop database import tool”**
 - Open-source Apache project
 - Originally developed at Cloudera
 - Included in CDH
- **Designed to import data from RDBMSs (Relational Database Management Systems) into HDFS**
 - Can also send data from HDFS to an RDBMS
- **Supports importing to and exporting from many Hadoop file types**
 - Hive tables
 - Avro files
 - HBase tables
 - Accumulo tables
- **Uses JDBC (Java Database Connectivity) to connect to the RDBMS**

How Does Sqoop Work?

- Sqoop examines each table and automatically generates a Java class to import data into HDFS
- It then creates and runs a Map-only MapReduce job to import the data
 - By default, four Mappers connect to the RDBMS
 - Each imports a quarter of the data



Sqoop Features

- **Imports a single table, or all tables in a database**
- **Can specify which rows to import**
 - Via a WHERE clause
- **Can specify which columns to import**
- **Can provide an arbitrary SELECT statement**
- **Sqoop can automatically create a Hive table based on the imported data**
- **Supports incremental imports of data**
- **Can export data from HDFS to a database table**

Sqoop Connectors

- **Custom Sqoop *connectors* exist for higher-speed import from some RDBMSs and other systems**
 - Use a system's native protocols to access data rather than JDBC
 - Provides much faster performance
 - Typically developed by the third-party RDBMS vendor
 - Sometimes in collaboration with Cloudera
- **Current systems supported by custom connectors include:**
 - Netezza
 - Teradata
 - Oracle Database (connector developed with Quest Software)
- **Others are in development**
- **Custom connectors are often not open source, but are free**

Sqoop Usage Examples

- **List all databases**

```
sqoop list-databases --username fred -P \
--connect jdbc:mysql://dbserver.example.com/
```

- **List all tables in the world database**

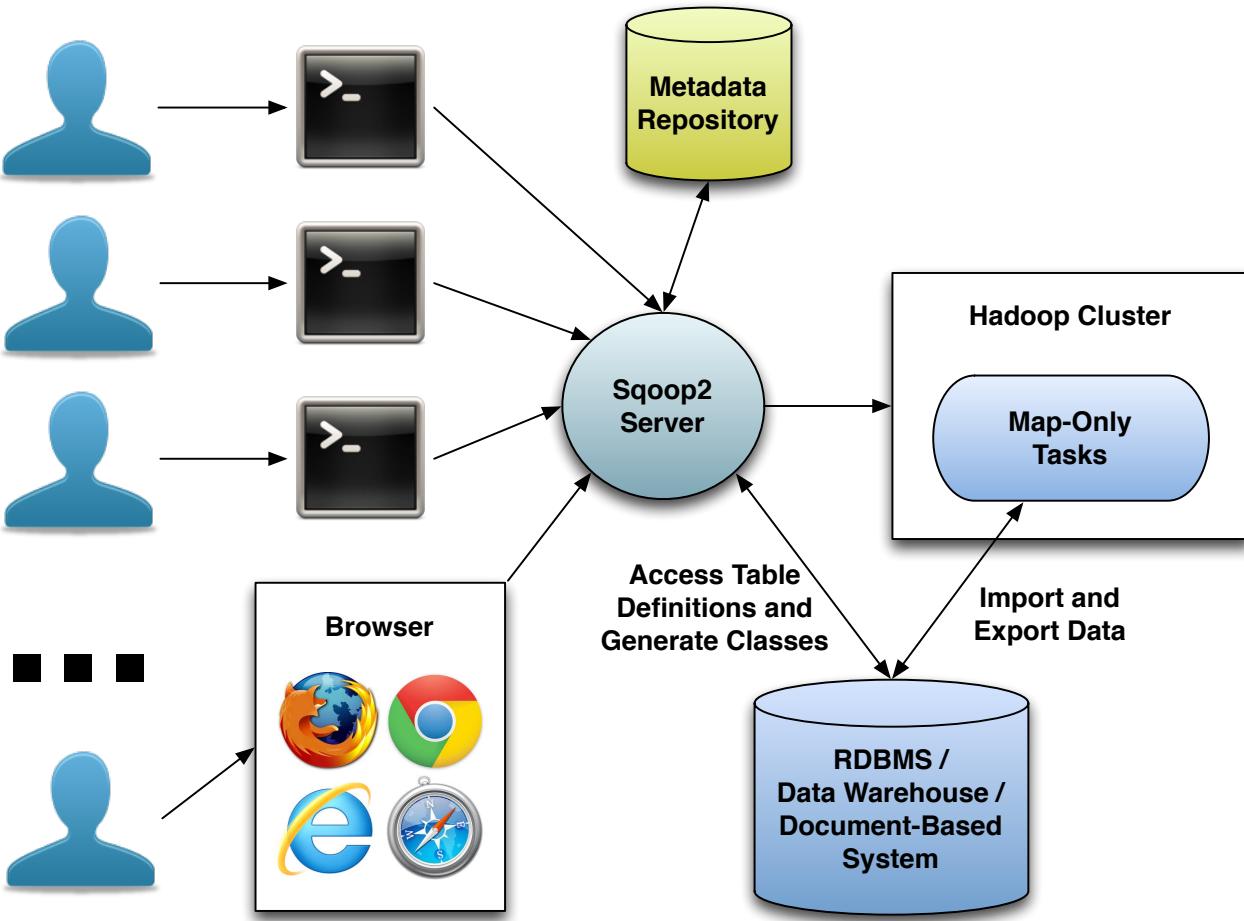
```
sqoop list-tables --username fred -P \
--connect jdbc:mysql://dbserver.example.com/world
```

- **Import all tables in the world database**

```
sqoop import-all-tables --username fred --password derf \
--connect jdbc:mysql://dbserver.example.com/world
```

Sqoop 2 – Sqoop as a Service (1)

- New version of Sqoop can be run as a service on a centrally-available machine



Sqoop 2 – Sqoop as a Service (2)

Functionality	Sqoop	Sqoop2
Installation and Configuration	<ul style="list-style-type: none">• Connectors and JDBC drivers are installed on every client• Database connectivity required for every client	<ul style="list-style-type: none">• Connectors and JDBC drivers are installed on the Sqoop2 server• Requires database connectivity for the Sqoop2 server
Client Interface	CLI only	CLI, Web UI, REST
Security	Every invocation requires credentials to RDBMS	Administrator specifies credentials when creating server-side Connection objects
Resource Management	No resource management	Administrator can limit the number of connections to the RDBMS

Sqoop 2 Status

- **Sqoop 2 is being actively developed**
 - It began shipping (alongside Sqoop) starting in CDH 4.2
- **Sqoop 2 is not yet at feature parity with Sqoop**
 - Implemented features are regarded as stable
 - Consider using Sqoop 2 unless you require a feature it lacks
- **We use Sqoop, rather than Sqoop 2, in this class**

Chapter Topics

Getting Data Into HDFS

- Ingesting Data From External Sources With Flume
- Hands-On Exercise: Using Flume
- Ingesting Data From Relational Databases With Sqoop
- **REST Interfaces**
- Best Practices for Importing Data
- Conclusion
- Hands-On Exercise: Importing Data With Sqoop

The WebHDFS REST API

- **Provides an HTTP/HTTPS REST interface to HDFS**
 - Supports both reads and writes from/to HDFS
 - Can be accessed from within a program or script
 - Can be used via command-line tools such as `curl` or `wget`
- **Installs with the HDFS service in Cloudera Manager**
 - Enabled by default (`dfs.webhdfs.enabled`)
- **Requires client access to every DataNode in the cluster**
- **Does not support HDFS HA deployments**

REST: REpresentational State Transfer

The HttpFS REST API

- **Provides an HTTP/HTTPS REST interface to HDFS**
 - The interface is identical to the WebHDFS REST interface
- **Installable part of the HDFS service in Cloudera Manager**
 - Choose to install it when install HDFS or “Add Role Instances” from existing deployed HDFS service
 - Installs and configures an HttpFS server
 - Enables proxy access to HDFS for an `httpfs` user
- **Requires client access to the HttpFS server only**
 - The HttpFS server then accesses HDFS
- **Supports HDFS HA deployments**

WebHDFS/HttpFS REST Interface Examples

- These examples will work with either WebHDFS or HttpFS
 - For WebHDFS, specify the NameNode host and port (default: 50070)
 - For HttpFS, specify the HttpFS server and port (default: 14000)
- Open and get the **shakespeare.txt** file
 - curl -i -L "http://host:port/webhdfs/v1/tmp/\shakespeare.txt?op=OPEN&user.name=training"
- Make the **mydir** directory
 - curl -i -X PUT "http://host:port/webhdfs/v1/user/\training/mydir?op=MKDIRS&user.name=training"

Chapter Topics

Getting Data Into HDFS

- Ingesting Data From External Sources With Flume
- Hands-On Exercise: Using Flume
- Ingesting Data From Relational Databases With Sqoop
- REST Interfaces
- **Best Practices for Importing Data**
- Conclusion
- Hands-On Exercise: Importing Data With Sqoop

What Do Others See As Data Is Imported?

- When a client starts to write data to HDFS, the NameNode marks the file as existing, but being of zero size
 - Other clients will see that as an empty file
- After each block is written, other clients will see that block
 - They will see the file growing as it is being created, one block at a time
- This is typically not a good idea
 - Other clients may begin to process a file as it is being written

Importing Data: Best Practices

- **Best practice is to import data into a temporary directory**
- **After the file is completely written, move data to the target directory**
 - This is an atomic operation
 - Happens very quickly since it merely requires an update of the NameNode's metadata
- **Many organizations standardize on a directory structure such as**
 - /incoming/<import_job_name>/<files>
 - /for_processing/<import_job_name>/<files>
 - /completed/<import_job_name>/<files>
- **It is the job's responsibility to move the files from for_processing to completed after the job has finished successfully**
- **Discussion point: your best practices?**

Chapter Topics

Getting Data Into HDFS

- Ingesting Data From External Sources With Flume
- Hands-On Exercise: Using Flume
- Ingesting Data From Relational Databases With Sqoop
- REST Interfaces
- Best Practices for Importing Data
- **Conclusion**
- Hands-On Exercise: Importing Data With Sqoop

Essential Points

- You can install Flume agents on systems such as Web servers and mail servers to extract, optionally transform, and pass data into HDFS
 - Flume scales extremely well and is in production use at many large organizations
- Flume uses the terms source, sink, and channel to describe its actors
 - A source is where an agent receives data from
 - A sink is where an agent sends data to
 - A channel is a queue between a source and a sink
- Using Sqoop, you can import data from a relational database into HDFS
- A REST interface is available for accessing HDFS
 - To use the REST interface, you must have enabled WebHDFS or deployed HttpFS
 - The REST interface is identical whether you use WebHDFS or HttpFS

Chapter Topics

Getting Data Into HDFS

- Ingesting Data From External Sources With Flume
- Hands-On Exercise: Using Flume
- Ingesting Data From Relational Databases With Sqoop
- REST Interfaces
- Best Practices for Importing Data
- Conclusion
- **Hands-On Exercise: Importing Data With Sqoop**

Hands-On Exercise: Importing Data with Sqoop

- In this Hands-On Exercise, you will import data from a relational database using Sqoop
- Please refer to the Hands-On Exercise Manual



Planning Your Hadoop Cluster

Chapter 8



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- **Planning Your Hadoop Cluster**
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Planning Your Hadoop Cluster

In this chapter, you will learn:

- **What issues to consider when planning your Hadoop cluster**
- **What types of hardware are typically used for Hadoop nodes**
- **How to optimally configure your network topology**
- **How to select the right operating system and Hadoop distribution**
- **How to plan for cluster management**

Chapter Topics

Planning Your Hadoop Cluster

- **General Planning Considerations**
- Choosing the Right Hardware
- Virtualization Options
- Network Considerations
- Configuring Nodes
- Conclusion

Basic Cluster Configuration

Master Nodes

HDFS
NameNode

HDFS
Secondary
NameNode*

YARN
Resource Manager

YARN
Standby
Resource Manager

YARN NodeManager
HDFS DataNode

* in this configuration, HDFS is not highly available. HA details discussed later.

Thinking About the Problem

- **Hadoop can run on a single machine**
 - Great for testing, developing
 - Obviously not practical for large amounts of data
- **Many people start with a small cluster and grow it as required**
 - Perhaps initially just four or six nodes
 - As the volume of data grows, more nodes can easily be added
- **Ways of deciding when the cluster needs to grow**
 - Increasing amount of computation power needed
 - Increasing amount of data which needs to be stored
 - Increasing amount of memory needed to process tasks

Cluster Growth Based on Storage Capacity

- **Basing your cluster growth on storage capacity is often a good method**
- **Example:**
 - Data grows by approximately 3TB per week
 - HDFS set up to replicate each block three times
 - Therefore, 9TB of extra storage space required per week
 - Plus some overhead – say, 25% of all disk space
 - Assuming machines with 16 x 3TB hard drives, this equates to a new machine required every four weeks
 - Alternatively: Two years of data – 1.2 petabytes – will require approximately 35 machines

Chapter Topics

Planning Your Hadoop Cluster

- General Planning Considerations
- **Choosing the Right Hardware**
- Virtualization Options
- Network Considerations
- Configuring Nodes
- Conclusion

Classifying Nodes

- **Nodes can be classified as either ‘worker nodes’ or ‘master nodes’**
- **Worker nodes run DataNode, NodeManager, and Impala Server daemons**
- **Master nodes run either a NameNode daemon, a Standby NameNode (or Secondary NameNode) Daemon, or a ResourceManager daemon**
 - On smaller clusters, NameNode and ResourceManager are often run on the same machine
 - Sometimes even Secondary NameNode is on the same machine as the NameNode and ResourceManager
 - Important that at least one copy of the NameNode’s metadata is stored on a separate machine in case of catastrophic failure

Worker Nodes – Recommended Configurations

- **Typical configurations for worker nodes**

- Midline – deep storage, 1Gb Ethernet
 - 16 x 3TB SATA II hard drives, in a non-RAID, JBOD* configuration
 - 1 or 2 of the 16 drives for the OS, with RAID-1 mirroring
 - 2 x 6-core 2.9GHz CPUs, 15MB cache
 - 256GB RAM
 - 2x1 Gigabit Ethernet
 - High-end – high memory, spindle dense, 10Gb Ethernet
 - 24 x 1TB Nearline/MDL SAS hard drives, in a non-RAID, JBOD* configuration
 - 2 x 6-core 2.9GHz CPUs, 15MB cache
 - 512GB RAM (or more)
 - 1x10 Gigabit Ethernet

* JBOD: Just a Bunch Of Disks

Worker Nodes – CPU

- Hex- and octo-core CPUs are commonly available
- Hyper-threading and quick-path interconnect (QPI) should be enabled
- Hadoop nodes are typically disk- and network-I/O bound
 - Therefore, top-of-the-range CPUs are usually not necessary
- Some types of Hadoop jobs do make heavy use of CPU resources
 - Clustering and classification
 - Complex text mining
 - Natural language processing
 - Feature extraction
 - Image manipulation
- You might need more processing power on your worker nodes if your specific workload requires it

Worker Nodes – RAM (1)

- Worker node configuration specifies the amount of memory and number of cores that Map tasks, Reduce tasks, and ApplicationMasters can use on that node
- Each Map and Reduce task typically takes 2GB to 4GB of RAM
- Each ApplicationMaster typically takes 1GB of RAM
- Worker nodes should *not* be using virtual memory
- Ensure you have enough RAM to run all tasks, plus overhead for the DataNode and NodeManager daemons, plus the operating system
- Rule of thumb:
 - Total number of tasks = Number of physical processor cores minus one
 - This is a starting point, and should not be taken as a definitive setting for all clusters

Worker Nodes – RAM (2)

- **New, memory-intensive processing frameworks are being deployed on many Hadoop clusters**
 - Impala
 - Spark
- **HDFS caching can also take advantage of extra RAM on worker nodes**
- **Good practice to equip your worker nodes with as much RAM as you can**
 - Memory configurations up to 512GB per worker node are not unusual for workloads with high memory requirements

Worker Nodes – Disk (1)

- **Hadoop's architecture impacts disk space requirements**
 - By default, HDFS data is replicated three times
 - Temporary data storage typically requires 20-30 percent of a cluster's raw disk capacity
- **In general, more spindles (disks) is better**
 - In practice, we see anywhere from four to 24 disks (or even more) per node
- **Use 3.5" disks**
 - Faster, cheaper, higher capacity than 2.5" disks
- **7,200 RPM SATA/SATA II drives are fine**
 - No need to buy 15,000 RPM drives
- **8 x 1.5TB drives is likely to be better than 6 x 2TB drives**
 - Different tasks are more likely to be accessing different disks

Worker Nodes – Disk (2)

- **A good practical maximum is 36TB per worker node**
 - More than that will result in massive network traffic if a node dies and block re-replication must take place
- **Recommendation: dedicate 1 disk for OS and logs, use the other disks for Hadoop data**
- **Mechanical hard drives currently provide a significantly better cost/ performance ratio than solid-state drives (SSDs)**
- **For hybrid clusters (both SSDs and HDDs), using SSDs for non-compressed intermediate shuffle data leads to significant performance gains**

Worker Nodes – Why Not RAID?

- **Worker nodes do not benefit from using RAID* storage**
 - HDFS provides built-in redundancy by replicating blocks across multiple nodes
 - RAID striping (RAID 0) is actually slower than the JBOD configuration used by HDFS
 - RAID 0 read and write operations are limited by the speed of the slowest disk in the RAID array
 - Disk operations on JBOD are independent, so the average speed is greater than that of the slowest disk
 - One test by Yahoo showed JBOD performing between 30% and 50% faster than RAID 0, depending on the operations being performed

* RAID: Redundant Array of Inexpensive Disks

What About Blade Servers?

- **Blade servers are not recommended**
 - Failure of a blade chassis results in many nodes being unavailable
 - Individual blades usually have very limited RAM and hard disk capacity
 - Network interconnection between the chassis and top-of-rack switch can become a bottleneck

Node Failure

- **Worker nodes are expected to fail at some point**
 - This is an assumption built into Hadoop
 - NameNode will automatically re-replicate blocks that were on the failed node to other nodes in the cluster, retaining the 3x replication requirement
 - ApplicationMasters will automatically re-assign tasks that were running on failed nodes
 - ResourceManagers will relaunch ApplicationMasters running on failed nodes
- **Master nodes are single points of failure if not configured for HA**
 - If the NameNode goes down, the cluster is inaccessible
 - If the ResourceManager goes down, no new jobs can run on the cluster
- **Configure the NameNode and ResourceManager for HA when running production workloads**

Master Node Hardware Recommendations

- **Carrier-class hardware**
- **Dual power supplies**
- **Dual Ethernet cards**
 - Bonded to provide failover
- **RAIDed hard drives**
- **Reasonable amount of RAM**
 - 64 GB for clusters of 20 nodes or less
 - 96 GB for clusters of up to 300 nodes
 - 128 GB for larger clusters

Chapter Topics

Planning Your Hadoop Cluster

- General Planning Considerations
- Choosing the Right Hardware
- **Virtualization Options**
- Network Considerations
- Configuring Nodes
- Conclusion

What About Virtualization?

- **If possible, we recommend dedicated physical hardware for your Hadoop cluster**
 - Perfectly reasonable to use virtualization where data center restrictions mean the use of dedicated hardware is not practical
- **If going the virtualization route:**
 - Benchmark the performance of bare-metal vs. virtual to understand any trade-offs specific to your setup
 - Shared storage can be a performance factor
 - Use local disks (multiple local disks even better)
 - Typically no way to guarantee rack placement of nodes
 - Consider the possibility that all three replicas of a block could end up on the same physical host
- **For cloud deployments, use Cloudera Director**

Deploying CDH on Microsoft Azure

- **Deploy Cloudera software on Microsoft Azure cloud infrastructure**
 - The supported OS is CentOS 6.5
 - Use the DS14 instance type with locally attached disks
 - 16 cores, 112GB RAM, 10 1-TB VHDs
 - Recommend use of Azure blob storage for HDFS disaster recovery
 - Choose between 4 and 20 worker nodes
 - Supported for Cloudera Manager 5 and CDH 5
 - Supports HDFS, YARN (MR2 Included), Spark, Impala, Cloudera Search, Hive, Pig, Oozie, ZooKeeper, and Crunch

Deploying CDH on VMware vSphere

- **Deploy Cloudera software on VMware vSphere infrastructure**
 - Cloudera software supported is nearly identical to physical deployments
- **Direct Attached HDFS Storage option**
 - VMs map physical disks to virtual disks (VMFS) or raw device mappings
- **Shared Isilon-based HDFS storage option**
 - With this option, the VMs run only the compute part on the cluster
- **Deployment details for both storage approaches are in the Reference Architecture documentation:**
 - <http://www.cloudera.com/content/cloudera/en/documentation/reference-architecture/latest.html>

Deploying CDH in the Cloud with Cloudera Director

- **Self-service provision and elastically scale clusters in cloud environments**
 - Amazon Web Services (AWS) or Google Cloud Platform (GCP)
- **Key Features**
 - A 100% open source Hadoop distribution
 - Supports complex cluster topologies
 - Long-lived and ephemeral cluster management
 - Provides a multi-cluster health dashboard
 - Cloudera Director Server UI, REST API, CLI tools
 - Supports CDH 4 and 5, Cloudera Manager 5
- **Example Deployment: Install Cloudera Director on an instance in AWS**
 - Create Amazon instance in VPC, install Cloudera Director on instance
 - Launch Cloudera Director, “Add Environment”, add cluster(s)
 - Cloudera Director distributes CDH parcel
- **Manage each Cloudera Director deployment with Cloudera Manager**

Chapter Topics

Planning Your Hadoop Cluster

- General Planning Considerations
- Choosing the Right Hardware
- Virtualization Options
- **Network Considerations**
- Configuring Nodes
- Conclusion

General Network Considerations (1)

- **Hadoop is very bandwidth-intensive!**
 - Often, all nodes are communicating with each other at the same time
- **Use dedicated switches for your Hadoop cluster**
- **Nodes are connected to a top-of-rack switch**
- **Nodes should be connected at a minimum speed of 1Gb/sec**
- **Consider 10Gb/sec connections in the following cases:**
 - Clusters storing very large amounts of data
 - Clusters in which typical jobs produce large amounts of intermediate data

General Network Considerations (2)

- Racks are interconnected via core switches
- Core switches should connect to top-of-rack switches at 10Gb/sec or faster
- Beware of oversubscription in top-of-rack and core switches
- Consider bonded Ethernet to mitigate against failure
- Consider redundant top-of-rack and core switches

Hostname Resolution

- When configuring Hadoop, you will be required to identify various nodes in Hadoop's configuration files
- Use hostnames, not IP addresses, to identify nodes when configuring Hadoop
- DNS is preferred for hostname resolution (rather than /etc/hosts)
 - Set hostnames to fully-qualified domain name (FQDN)
 - Each host must be able to:
 - Perform a forward lookup on its own hostname
 - Perform a reverse lookup using its own ip address
- Forward and reverse lookups must work correctly whether you are using DNS or /etc/hosts for name resolution
 - If the results do not match, major problems can occur

Chapter Topics

Planning Your Hadoop Cluster

- General Planning Considerations
- Choosing the Right Hardware
- Virtualization Options
- Network Considerations
- **Configuring Nodes**
- Conclusion

Operating System Recommendations (1)

- **Choose a distribution that you are comfortable administering**
- **CentOS: geared towards servers rather than individual workstations**
 - Conservative about package versions
 - Very widely used in production
- **RedHat Enterprise Linux (RHEL): RedHat-supported analog to CentOS**
 - Includes support contracts, for a price
- **In production, we often see a mixture of RHEL and CentOS machines**
 - Often RHEL on master nodes, CentOS on workers

Operating System Recommendations (2)

- **Ubuntu: Very popular distribution, based on Debian**
 - Both desktop and server versions available
 - Use an LTS (Long Term Support) version
- **SuSE: popular distribution, especially in Europe**
 - Cloudera provides CDH packages for SuSE

Configuring The System (1)

- **Do not use Linux's LVM (Logical Volume Manager) to make all your disks appear as a single volume**
 - As with RAID 0, this limits speed to that of the slowest disk
 - Can also result in the loss of all data on the node if a single disk fails
- **Check the machines' BIOS* settings**
 - BIOS settings may not be configured for optimal performance
 - For example, if you have SATA drives make sure IDE emulation is not enabled
- **Test disk I/O speed with `hdparm -t`**
 - Example:
`hdparm -t /dev/sda1`
 - You should see speeds of 70MB/sec or more
 - Anything less is an indication of possible problems

* BIOS: Basic Input/Output System

Configuring The System (2)

- **Hadoop has no specific disk partitioning requirements**
 - Use whatever partitioning system makes sense to you
- **Mount disks with the `noatime` option**
- **Common directory structure for data mount points:**
 - /data/<n>/dfs/nn
 - /data/<n>/dfs/dn
 - /data/<n>/dfs/snn
 - /data/<n>/mapred/local
- **Reduce the *swappiness* of the system in `/etc/sysctl.conf`**
 - Set `vm.swappiness` to 0 unless using RHEL kernel 2.6.32-303 or later
- **Configure IPTables if required by your security policies – Hadoop requires many ports for communication**
 - From Cloudera Manager's Cluster page, choose *Configuration > All Port Configurations* to see all ports used

Configuring The System (3)

- **Disable Transparent Huge Page compaction**

- Can degrade the performance of Hadoop workloads
- Open the defrag file of your OS to see if it is enabled
 - Red Hat/CentOS: /sys/kernel/mm/\
redhat_transparent_hugepage/defrag
 - Ubuntu/Debian, OEL, SLES: /sys/kernel/mm/\
transparent_hugepage/defrag
- A line reading '[always] never' means it is enabled
- A line reading '[never] always' means it is disabled
- To temporarily disable it
 - `sudo sh -c "echo 'never' > defrag_file_pathname"`
- Add the following to /etc/rc.local to persist the change
 - `echo never > defrag_file_pathname`

Filesystem Considerations

- **Cloudera recommends the ext3 and ext4 filesystems**
 - ext4 is more commonly used on new clusters
- **XFS provides some performance benefit during kickstart**
 - It formats in 0 seconds, vs. several minutes for each disk with ext3/ext4
- **Currently, more testing is done at Cloudera on ext4 than XFS**

Operating System Parameters

- **Increase the `nofile` ulimit for the `cloudera-scm` user to at least 32K**
 - Cloudera Manager sets this to 32K in `/usr/sbin/cmf-agent` by default
- **Disable IPv6**
- **Disable SELinux if possible**
 - Incurs a performance penalty on a Hadoop cluster
 - Configuration is non-trivial
- **Install and configure the `ntp` daemon**
 - Ensures the time on all nodes is synchronized
 - Important for HBase, ZooKeeper, Kerberos
 - Useful when using logs to debug problems

Java Virtual Machine (JVM) Requirements

- **Always use the official Oracle JDK (<http://java.com/>)**
 - Hadoop is complex software, and often exposes bugs in other JDK implementations
- **Version 1.7 is required**
 - CDH5 is certified with 1.7.0_55
 - Any later maintenance release should be acceptable for production
 - Refer to the CDH Release Notes for details
- **CDH 5 does not support version 1.6**

The Cloudera Manager Host Inspector

- The Host Inspector checks for many of the items just discussed

- Validates select OS settings, networking settings, system time, user and group settings, and component versions

The screenshot shows the Cloudera Manager interface with the 'Hosts' tab selected. A red box highlights the 'Host Inspector' button in the top right corner of the status bar. A red arrow points from this button down to the 'Inspector Results' section. This section contains a 'Validations' table with the following data:

Validation	Status
Inspector ran on all 8 hosts.	Success
The following failures were observed in checking hostnames...	Failure
The following errors were found while looking for conflicting init scripts. Use 'chkconfig' to disable init scripts to avoid conflicts with daemons managed by Cloudera Manager. Typically, you may be able to continue with installation, but, after a reboot, processes may fail to start because of a conflict. >	Warning
No errors were found while checking /etc/hosts.	Success
All hosts resolved localhost to 127.0.0.1.	Success
All hosts checked resolved each other's hostnames correctly and in a timely manner.	Success
Host clocks are approximately in sync (within ten minutes).	Success
Host time zones are consistent across the cluster.	Success
No users or groups are missing.	Success
No conflicts detected between packages and parcels.	Success
No kernel versions that are known to be bad are running.	Success
Cloudera recommends setting /proc/sys/vm/swappiness to 0. Current setting is 10. Use the sysctl command to change this setting at runtime and edit /etc/sysctl.conf for this setting to be saved after a reboot. You may continue with installation, but you may run into issues with Cloudera Manager reporting that your hosts are unhealthy because they are swapping. The following hosts are affected: >	Warning
Transparent Huge Pages is enabled and can cause significant performance problems. Kernel with release 'CentOS release 6.2 (Final)' and version '2.6.32-220.7.1.el6.x86_64' has enabled set to '[always] never' and defrag set to '[always] never'. Run "echo never > /sys/kernel/mm/redhat_transparent_hugepage/defrag" to disable this, then add the same command to an init script such as /etc/rc.local so it will be set upon system reboot. Alternatively, upgrade to RHEL 6.5 or later, which does not have this bug. The following hosts are affected: >	Warning

Chapter Topics

Planning Your Hadoop Cluster

- General Planning Considerations
- Choosing the Right Hardware
- Virtualization Options
- Network Considerations
- Configuring Nodes
- **Conclusion**

Essential Points

- **Master nodes run the NameNode, Standby NameNode (or Secondary NameNode), and ResourceManager**
 - Provision with carrier-class hardware and lots of RAM
- **Worker nodes run DataNodes and NodeManagers**
 - Provision with industry-standard hardware
 - Consider your data storage growth rate when planning current and future cluster size
- **RAID (on worker nodes) and virtualization can incur a performance penalty**
- **Make sure that forward and reverse domain lookups work when configuring a cluster**



Installing and Configuring Hive, Impala, and Pig

Chapter 9



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- **Installing and Configuring Hive, Impala, and Pig**
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Installing and Configuring Hive, Impala, and Pig

In this chapter, you will learn:

- **Hive features and basic configuration**
- **Impala features and basic configuration**
- **Pig features and installation**

Note

- Note that this chapter does not go into any significant detail about Hive, Impala, or Pig
- Our intention is to draw your attention to issues System Administrators will need to deal with, if users request these products be installed
 - The *Cloudera Data Analyst Training: Using Pig, Hive, and Impala with Hadoop* course provides detailed information about how to use these three components

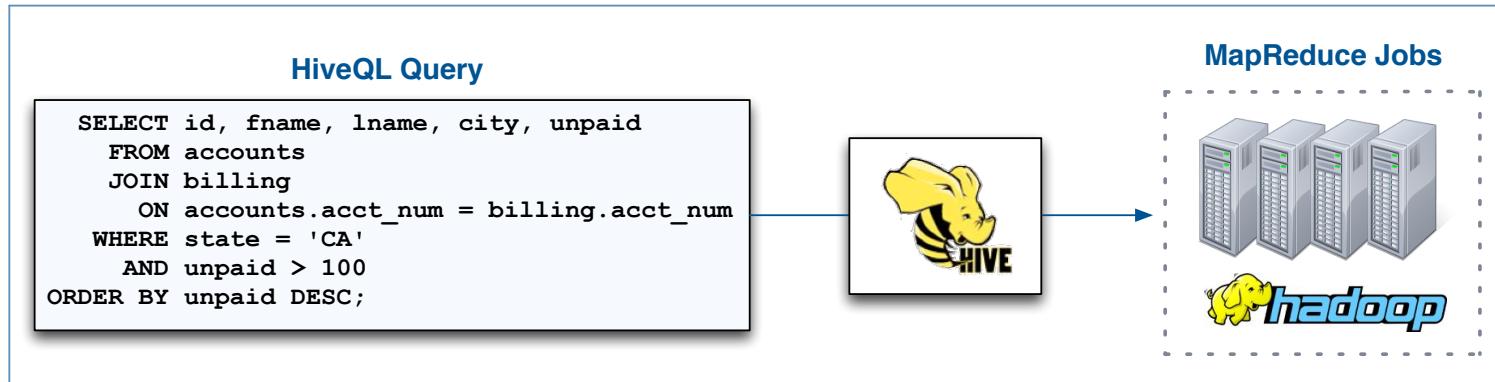
Chapter Topics

Installing and Configuring Hive, Impala, and Pig

- **Hive**
- Impala
- Pig
- Conclusion
- Hands-On Exercise: Querying HDFS With Hive and Cloudera Impala

What is Hive?

- **Hive is a high-level tool that uses an SQL-like syntax to query HDFS data**
 - Fulfils queries by running MapReduce jobs on the cluster
 - Automatically runs the jobs, and displays the results to the user
 - Hive is an open source Apache project originally developed at Facebook



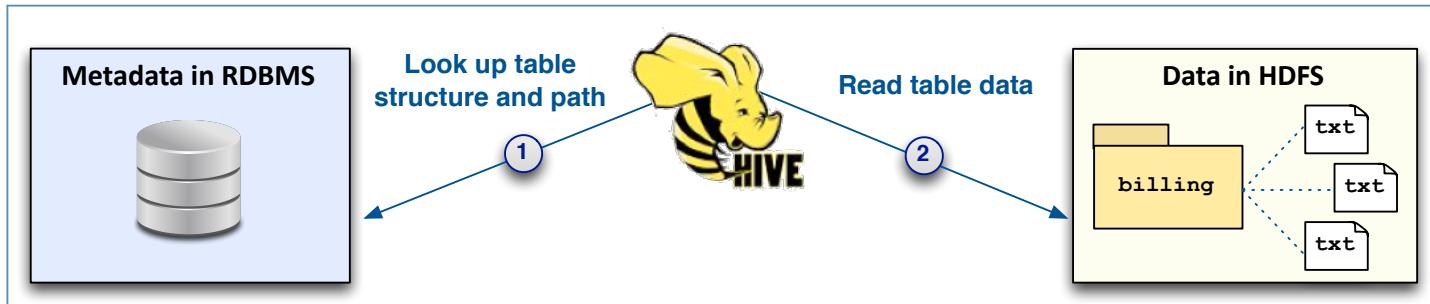
- **Motivation**
 - Many data analysts are familiar with Structured Query Language (SQL)
 - SQL is the *de facto* standard for querying data in Relational Database Management Systems (RDBMSs)
 - Data analysts often less familiar with programming languages (e.g. Java)

How Hive Works

- A Hive table is a directory of data in HDFS with associated metadata
- Tables are created by describing pre-existing data in HDFS

```
CREATE TABLE products (
    id INT,
    name STRING,
    price INT
);
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t';
```

- *Metadata (table structure and path to data) is stored in an RDBMS*



Hive Tables

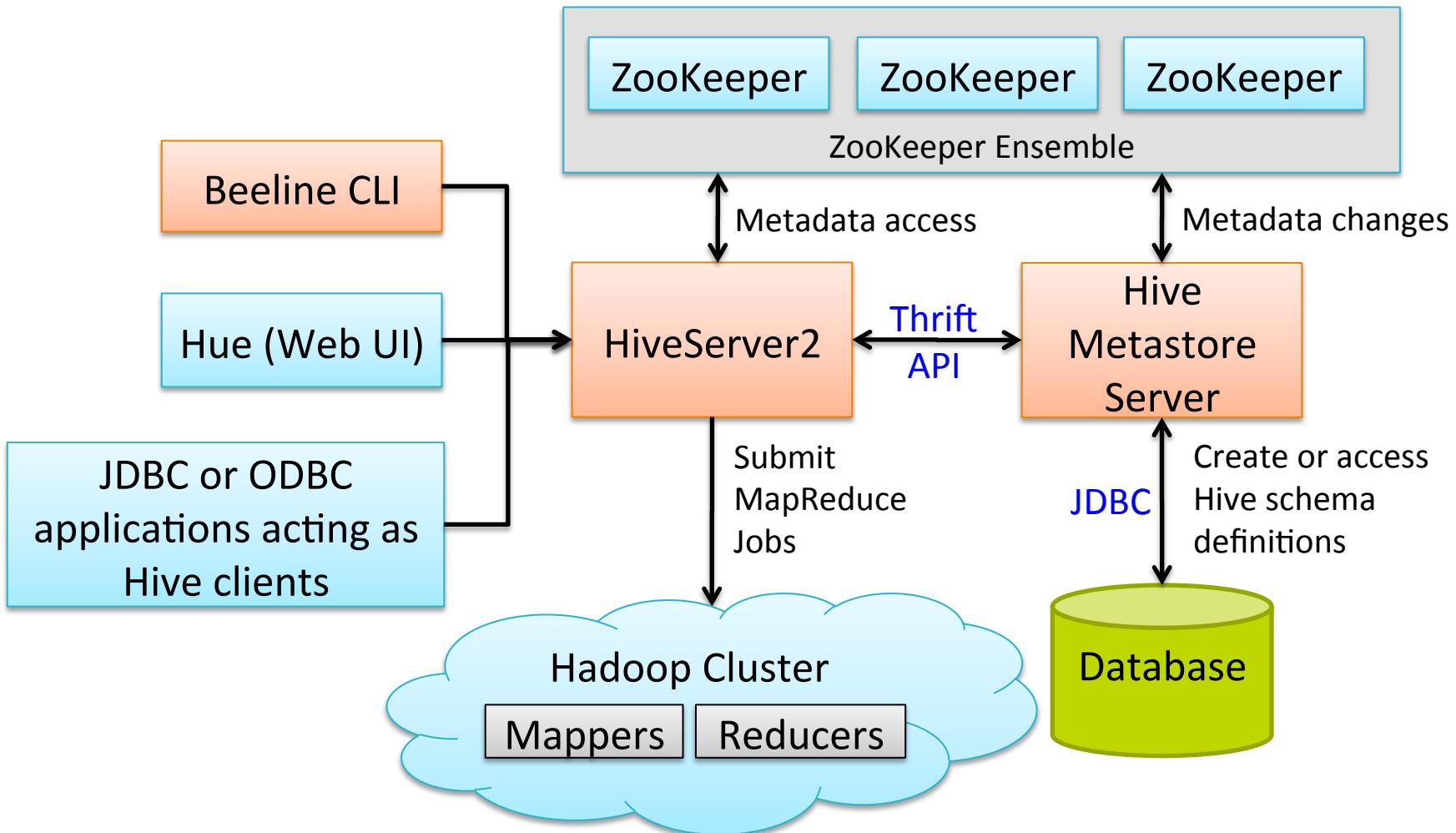
- **A Hive Table represents an HDFS directory**
 - Hive interprets all files in the directory as the contents of the table
 - Hive stores information about how the rows and columns are delimited within the files in the *Hive Metastore*
- **Tables are created as either managed or external**
 - Managed
 - Hive manages underlying data files in its HDFS warehouse directory
 - Default location is /user/hive/warehouse
 - If the table is dropped, the schema and the HDFS data is deleted
 - External
 - Reference the source data directory when defining the table
 - Data files queried from their original location in HDFS
 - If the table is dropped, only the table schema is deleted

Hive Is Not an RDBMS

- Note that Hive is *not* an RDBMS!

- Results take several seconds, minutes, or even hours to be produced
 - Not possible to modify the data using HiveQL
 - UPDATE and DELETE are *not* supported

Hive Basic Architecture



Hive Service Roles Installed by Cloudera Manager

- **Hive Metastore Server**

- Manages the remote Metastore process

- **HiveServer2**

- Supports the Thrift API used by clients
 - Has a Hive CLI named Beeline

- **Hive Gateway**

- To deploy Hive client configurations to specific nodes in the cluster, add the Hive Gateway role instance to those nodes
 - Nodes with the Gateway role instance will receive the latest client configurations when you choose “Deploy Client Configuration” from Cloudera Manager
 - Client Configuration files can also be manually downloaded and distributed

Hive Metastore Server and Deployment Mode Options

- **Hive Metastore Server (Remote mode)**

- Recommended deployment mode
 - Datastore resides in a standalone RDBMS such as MySQL
 - HiveServer2 and other processes (e.g., Impala) communicate with the metastore service via JDBC
 - Advantage over Local mode:
 - Do not need to share JDBC login information for metastore database with each Hive user

- **Local mode**

- Functionality of Metastore Server embedded in the HiveServer process
 - Database runs separately, accessed via JDBC

- **Embedded mode**

- Supports only one active user – for experimental purposes only
 - Uses Apache Derby, a Java-based RDBMS

HiveServer2

- **HiveServer2 runs Hive as a server**
 - A container for the Hive execution engine
 - Enables remote clients to execute queries against Hive and retrieve the results
 - Accessible via JDBC, ODBC, or Thrift
 - Example clients: Beeline (the Hive CLI), Hue (Web UI)
- **Supports concurrent queries from more than one Hive client**
 - Why is concurrency support needed?
 - Example: a Hive client issues a DROP TABLE command, while at the same time another Hive client running on a different machine runs a SELECT query against the same table
 - Hive concurrency requires ZooKeeper to guard against data corruption

HiveServer2 Uses ZooKeeper

- **ZooKeeper is a distributed coordination service for Hadoop**
 - Allows processes to share information with each other in a reliable and redundant way
- **Used for many purposes**
 - Examples: Leader election, distributed synchronization, failure recovery
- **ZooKeeper always runs on a number of machines**
 - Called a “ZooKeeper ensemble”
 - Makes the service highly available
- **HiveServer2 uses ZooKeeper to support concurrent clients**
 - Client processes could be running on different machines
- **Other Hadoop components can share the same ZooKeeper ensemble**

Installing and Starting a ZooKeeper Ensemble

- **Install a ZooKeeper ensemble on an odd number of hosts**
 - Typically 3 hosts
- **Each ZooKeeper server process should ideally have its own dedicated disk on the node on which it runs**
- **In Cloudera Manager, choose “Add a Service” for the cluster**
 - Choose ZooKeeper and install to 3 hosts
 - Start the ZooKeeper servers
- **If a majority of the ZooKeeper servers in the ensemble are available, then the ZooKeeper service will be available**

Installing and Configuring Hive (1)

1. Configure a connector to the remote RDBMS

- Example: sudo yum install mysql-connector-java

2. Create a metastore database in your RDBMS

- CREATE DATABASE metastore;

3. Create a database user with appropriate privileges

- USE metastore;

```
CREATE USER 'hiveuser'@'%' IDENTIFIED BY \
'password';
```

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM \
'hiveuser'@'%';
```

```
GRANT SELECT, INSERT, UPDATE, DELETE, \
LOCK TABLES, EXECUTE, CREATE, ALTER \
ON metastore.* TO 'hiveuser'@'%';
```

Installing and Configuring Hive (2)

- 3. Install ZooKeeper ensemble**
- 4. Run the ‘Add a Service’ wizard in Cloudera Manager**
 - Choose to install Hive
 - Specify ZooKeeper as a prerequisite
 - Point to the metastore database
 - Cloudera Manager Installs the Hive service
 - Creates Hive Metastore database tables in the RDBMS you specify
 - Creates Hive user and warehouse directories in HDFS
 - Installs Hive Metastore Server and HiveServer2 roles
 - Deploys client configuration files to Hive Gateway node(s)

Using the Beeline CLI

- Connect Beeline to HiveServer2
- Define Hive tables or run HiveQL queries

```
[training@elephant ~]$ beeline -u jdbc:hive2://elephant:10000 -n training
scan complete in 4ms
Connecting to jdbc:hive2://elephant:10000
Connected to: Apache Hive (version 0.12.0-cdh5.0.0)
Driver: Hive JDBC (version 0.12.0-cdh5.0.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 0.12.0-cdh5.0.0 by Apache Hive
0: jdbc:hive2://elephant:10000> SELECT COUNT(*) FROM movierating;
+-----+
| _c0   |
+-----+
| 1000205 |
+-----+
1 row selected (83.514 seconds)
0: jdbc:hive2://elephant:10000> !quit
Closing: org.apache.hive.jdbc.HiveConnection
[training@elephant ~]$
```

Chapter Topics

Installing and Configuring Hive, Impala, and Pig

- Hive
- **Impala**
- Pig
- Conclusion
- Hands-On Exercise: Querying HDFS With Hive and Cloudera Impala

What Is Impala?

- **A 100% open source project created at Cloudera**
 - Uses the same Apache software license as Hadoop itself
- **Like Hive, Impala allows users to query data in HDFS using an SQL-like language**
- **Unlike Hive, Impala does not turn queries into MapReduce jobs**
 - Impala queries run on an additional set of daemons that run on the Hadoop cluster
 - Often referred to as ‘Impala Servers’
 - Impala queries run *significantly* faster than Hive queries
 - Tests show improvements of 10x to 50x or more
 - Impala also supports many simultaneous users much more efficiently than Hive
- **Impala uses the same shared Metastore that Hive uses**
 - Tables created in Hive are visible in Impala (and vice versa)



The Impala Shell

- The Impala Shell is an interactive tool similar to the Beeline shell for Hive
- Execute `impala-shell` command to start the shell, then run queries
 - Some log messages truncated to better fit the slide

```
$ impala-shell
Connected to elephant:21000
Server version: impalad version 2.1.0-cdh5
Welcome to the Impala shell.
[elephant:21000] > SELECT acct_num, first_name,
last_name FROM accounts WHERE zipcode='90210' ;
+-----+-----+-----+
| acct_num | first_name | last_name |
+-----+-----+-----+
| 6029     | Brian       | Ferrara   |
| 9493     | Mickey      | Kunkle    |
+-----+-----+-----+
> quit;
```

Components of Impala

- **Impala shares the Hive Metastore**
- ***Impala Daemon* – typically runs on each worker node in the cluster**
 - Co-located with the HDFS DataNode
 - Accepts queries from the impala-shell, Hue, JDBC, or ODBC
 - Checks the local metadata cache
 - Distributes the work to other Impala Daemons in the cluster
 - Daemons read and write to data files and stream results to the client
- **Two other daemons running on master nodes support query execution**
 - ***Impala State Store*** (one per cluster)
 - Provides lookup service and status checks for Impala daemons
 - ***Impala Catalog Server*** (one per cluster)
 - Relays metadata changes to all the Impala Daemons in a cluster

Installing Impala

- **To install Impala, run the ‘Add a Service’ wizard in Cloudera Manager**
 - Choose to install Impala
 - Specify HDFS and Hive as the prerequisites
 - Cloudera Manager Installs the Impala service
 - Installs Impala Catalog Server, Impala StateStore and the Impala Daemons
 - Creates Impala user directory in HDFS
 - Deploys the Impala Client (`impala-shell`) to all nodes in the cluster that have the Impala Daemon

Impala Performance Configuration

- Short-Circuit Reads are enabled in HDFS property settings when Impala is installed using Cloudera Manager
 - To find the setting, search for “shortcircuit”

The screenshot shows the Cloudera Manager interface for a cluster named "Cluster 1". The navigation bar includes Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. Under the Clusters section, there are links for HDFS, Status, Instances, Configuration, Commands, Audits, and File Browser. The Configuration tab is selected. A search bar at the top has the text "shortcircuit". Below the search bar, a message says "⚠ 1 validation warning below." A table lists a single configuration entry:

Category	Property	Value
Service-Wide / Performance	Enable HDFS Short Circuit Read dfs.client.read.shortcircuit	<input checked="" type="checkbox"/> default value

- Short-Circuit Reads improves Impala's performance
 - Impala can bypass the DataNode when reading data
 - Read HDFS file blocks directly

Impala Daemon Memory Configuration

- Configure the amount of system memory available to the Impala Daemon
 - Configure in Cloudera Manager’s Impala Configuration page
 - Run a search on “Impala Daemon Memory Limit”
 - Default value: (empty) – allows Impala to pick its own limit
 - Set as needed based on system activity, example: 64GB

Category	Property	Value
Impala Daemon Default Group / Resource Management	Impala Daemon Memory Limit mem_limit	Default value is empty. Click to edit. B

- The setting is ignored when Dynamic Resource Pools is enabled
 - Dynamic Resource Pools discussed in a later chapter

Impala Queries in Cloudera Manager

- In Cloudera Manager, choose “Impala Queries” from the Clusters menu
 - Options to filter queries and view query details

The screenshot shows the Cloudera Manager interface for the 'Impala' section. The top navigation bar includes 'Home', 'Clusters', 'Hosts', 'Diagnostics', 'Audits', 'Charts', 'Backup', and 'Administration'. The 'Clusters' tab is selected. Below the navigation is a search bar and a date range selector ('7 days preceding March 2 2015, 9:46 PM PST'). The main area has tabs for 'Status', 'Instances', 'Configuration', 'Commands', 'Audits', 'Queries' (which is selected), and 'Charts'. A green circular status indicator is present. Below these are sections for 'Workload Summary' (Completed Queries), 'Aggregate Peak Memory Usage' (8.1 MiB), 'Duration' (0ms-1s, 1s-2s, 5s-6s), and 'HDFS Bytes Read'. The 'Results' tab is active under the 'Queries' section, displaying a query log for 'select COUNT(*) FROM movierating' with details like User: training, Database: default, Query Type: QUERY, Coordinator: horse, Duration: 1.02s, Rows Produced: 1, Aggregate Peak Memory Usage: 8.1 MiB, HDFS Bytes Read: 11 MiB, and Threads: CPU Time: 2.28s. A red box highlights the 'Query Details' section, which lists 'User's Impala queries' and 'Queries in the same YARN pool'.

cloudera manager

5 Search (Hotkey: /) Support admin

Home Clusters Hosts Diagnostics Audits Charts Backup Administration

Cluster 1 7 days preceding March 2 2015, 9:46 PM PST

Impala Status Instances Configuration Commands Audits Queries Charts Actions

Queries 30m 1h 2h 6h 12h 1d 7d 30d

Search for Impala queries, e.g. 'rows_produced > 1000', or press space to start

Search Select Attributes Best Practices

Workload Summary (For Completed Queries)

Aggregate Peak Memory Usage 8.1 MiB

Duration 0ms - 1s 1s - 2s 5s - 6s

HDFS Bytes Read

Results Charts Export

02/24/2015 6:02 PM - 02/24/2015 6:02 PM

select COUNT(*) FROM movierating
User: training
Database: default
Query Type: QUERY
Coordinator: horse
Duration: 1.02s
Rows Produced: 1
Aggregate Peak Memory Usage: 8.1 MiB
HDFS Bytes Read: 11 MiB
Threads: CPU Time: 2.28s

Query Details

User's Impala queries
Queries in the same YARN pool

Chapter Topics

Installing and Configuring Hive, Impala, and Pig

- Hive
- Impala
- **Pig**
- Conclusion
- Hands-On Exercise: Querying HDFS With Hive and Cloudera Impala

What Is Pig?

- **Pig is a client-side application for data analysis and processing on Hadoop**
 - Open source Apache project originally created at Yahoo
- **Pig supports high-level data processing**
 - Alternative to writing low-level MapReduce code
 - Especially good at joining and transforming data
 - Like Hive, executes as MapReduce jobs
- **Provides a scripting language known as *Pig Latin* for processing data**
- **Use Pig interactively via the Grunt shell**

```
$ pig  
grunt> fs -ls;
```

Sample Pig Script

- A Pig Latin script consists of
 - One or more LOAD statements to read in data
 - Transformation statements to process the data
 - A DUMP statement or STORE statement to generate the output
- Example Pig Script

```
movies = LOAD '/data/films' AS  
        (id:int, name:chararray, year:int);  
  
ratings = LOAD '/data/ratings' AS  
        (movie_id: int, user_id: int, score:int);  
  
jnd = JOIN movies BY id, ratings BY movie_id;  
recent = FILTER jnd BY year > 1995;  
srted = ORDER recent BY name DESC;  
justafew = LIMIT srted 50;  
STORE justafew INTO '/data/pigoutput';
```

Installing Pig

- **Pig runs as a client-side application**
- **Pig is installed when the Cloudera Manager Agent is installed to a host and the CDH parcel is downloaded, distributed, and activated**
 - Even if no roles are installed to the host
- **Pig can also be installed standalone**
 - Example: `sudo yum install pig`

Chapter Topics

Installing and Configuring Hive, Impala, and Pig

- Hive
- Impala
- Pig
- **Conclusion**
- Hands-On Exercise: Querying HDFS With Hive, and Cloudera Impala

Essential Points

- **Hive is a high-level abstraction on top of MapReduce**
 - Runs MapReduce jobs on Hadoop based on HiveQL statements
- **Impala runs a separate set of daemons from MapReduce to process HiveQL queries**
 - One State Store Server, one Catalog Server
 - Impala Server daemons co-located with DataNodes
- **Hive and Impala use a common metastore to store metadata such as column names and data types**
 - For Hive, the metastore can be single-user
 - For Impala, the metastore must be shareable
- **Pig is another high-level abstraction on top of MapReduce**
 - Runs MapReduce jobs on Hadoop based on Pig Latin statements

Chapter Topics

Installing and Configuring Hive, Impala, and Pig

- Hive
- Impala
- Pig
- Conclusion
- **Hands-On Exercise: Querying HDFS With Hive and Cloudera Impala**

Hands-On Exercise: Querying HDFS With Hive and Cloudera Impala

- In this Hands-On Exercise, you will install and configure Hive and Impala on your cluster and run queries
- Please refer to the Hands-On Exercise Manual
- Cluster deployment after exercise completion:

				
elephant	✓			
horse		✓	✓	✓
monkey			✓	
tiger				✓
HDFS NameNode	✓			
HDFS SecondaryNameNode				✓
HDFS DataNode	✓	✓		
HDFS Balancer		✓		
YARN (MR2 Included) NodeManager	✓	✓	✓	✓
YARN (MR2 Included) ResourceManager		✓		
YARN (MR2 Included) JobHistory Server			✓	
ZooKeeper Server	✓	✓		✓
Sqoop 1 Client Gateway	✓			
Flume Agent	✓	✓		
HiveServer2	✓			
Hive Gateway	✓			
Hive Metastore Server	✓			
Impala Catalog Server			✓	
Impala StateStore			✓	
Impala Daemon	✓	✓	✓	✓



Hadoop Clients Including Hue

Chapter 10



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- **Hadoop Clients Including Hue**
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Hadoop Clients Including Hue

In this chapter, you will learn:

- **What Hadoop clients are**
- **How to install and configure Hadoop clients**
- **How to install and configure Hue**
- **How Hue authenticates and authorizes user access**

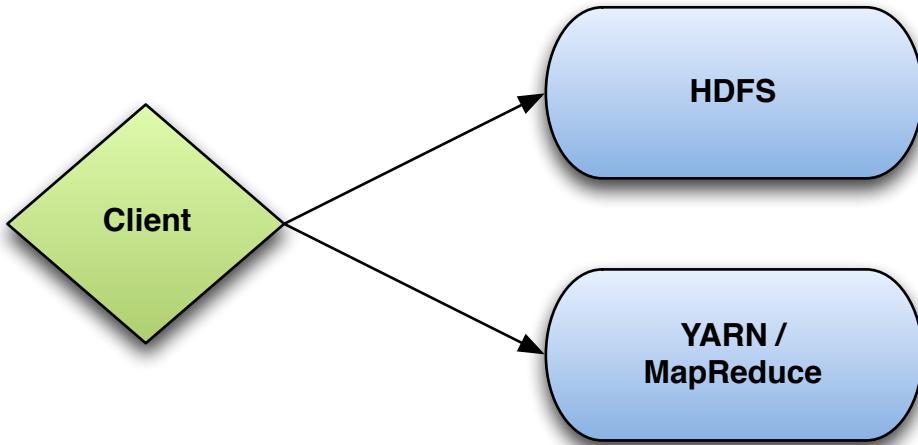
Chapter Topics

Hadoop Clients

- **What Are Hadoop Clients?**
- Installing and Configuring Hadoop Clients
- Installing and Configuring Hue
- Hue Authentication and Authorization
- Hands-On Exercise: Using Hue to Control Hadoop User Access
- Conclusion

Hadoop Client

Hadoop Client
accesses services on a
Hadoop Cluster using
the Hadoop APIs



Services running
on a Hadoop cluster

- A Hadoop client requires the Hadoop API and configuration to connect to Hadoop components

Examples of Hadoop Clients

- **Command-line Hadoop clients**

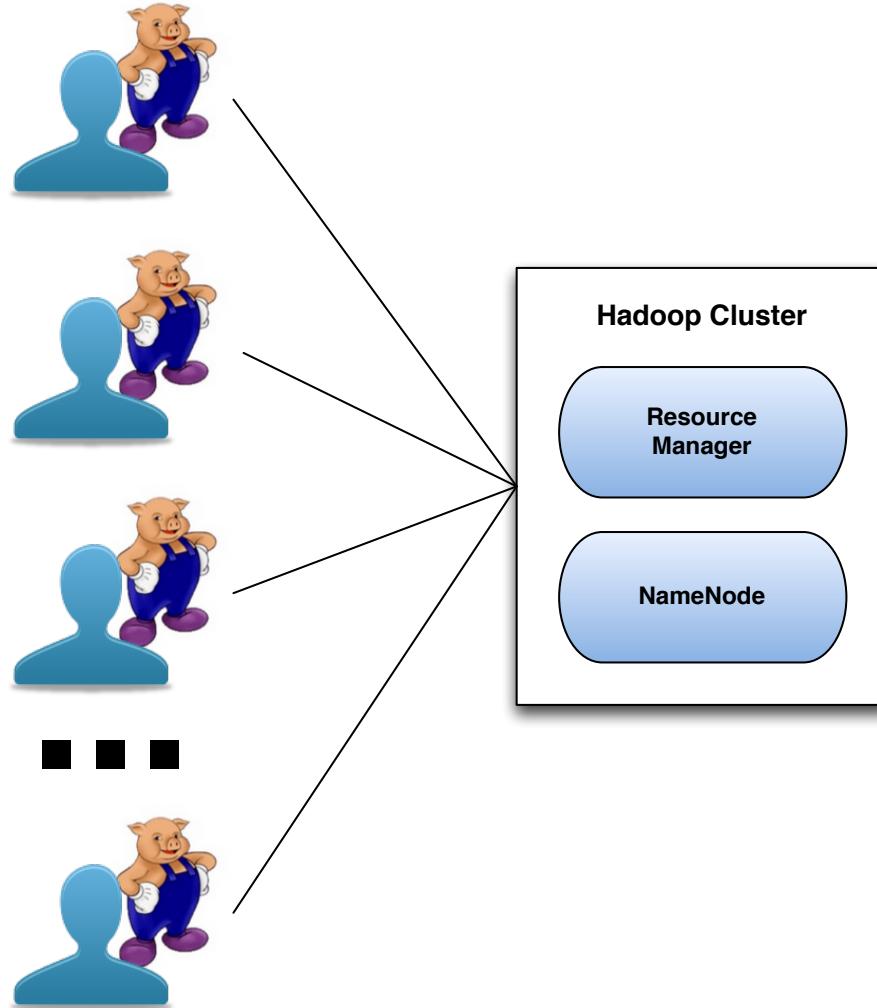
- The hadoop shell (hdfs dfs or hadoop fs)
- The Pig shell
- The Sqoop command-line interface

- **Server daemons acting as Hadoop clients**

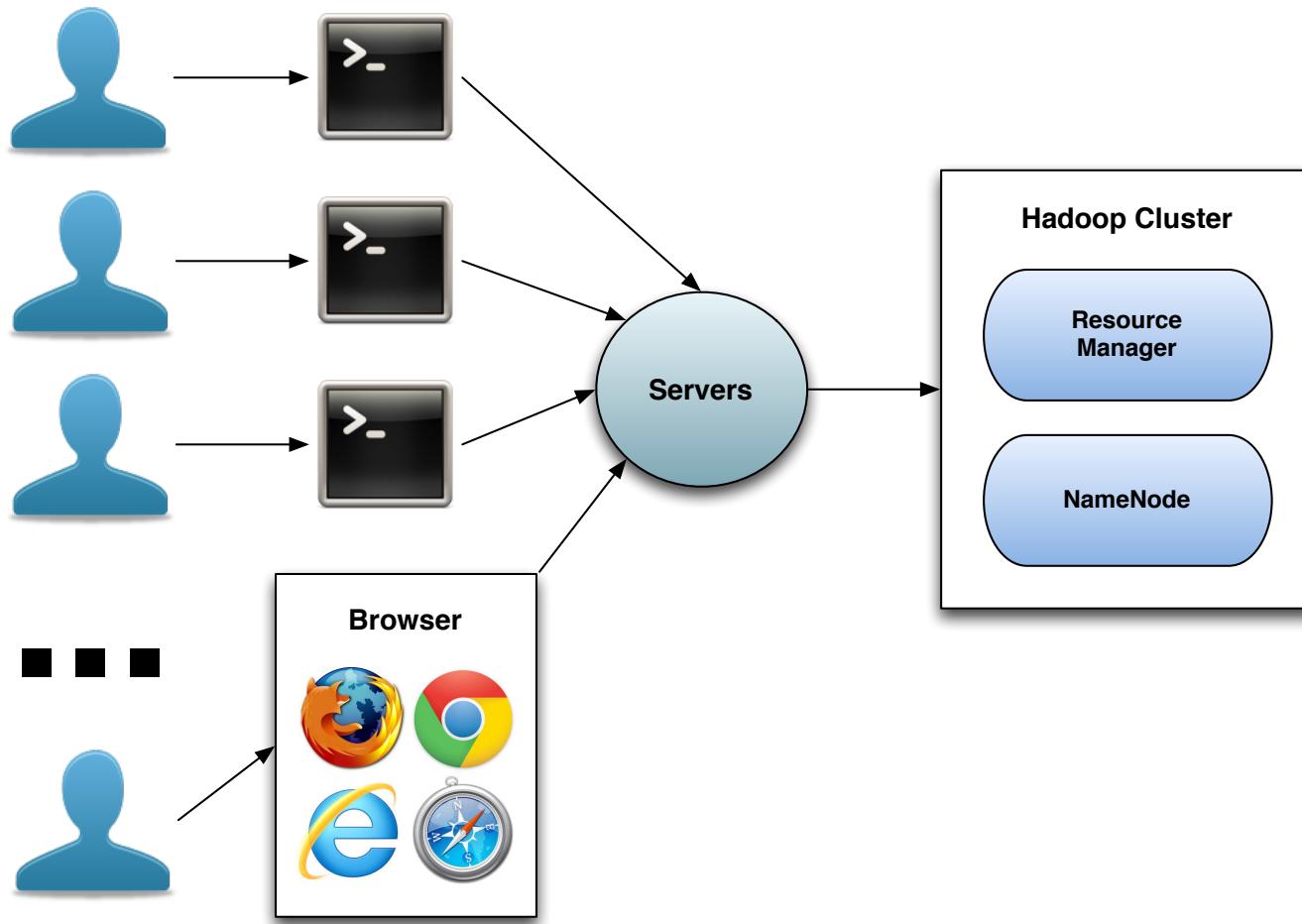
- Flume agent
- Centralized Sqoop and Hive servers
- Oozie

- **Map and Reduce tasks**

Deployment Example – End Users’ Systems as Hadoop Clients



Deployment Example – Servers as Hadoop Clients



Chapter Topics

Hadoop Clients

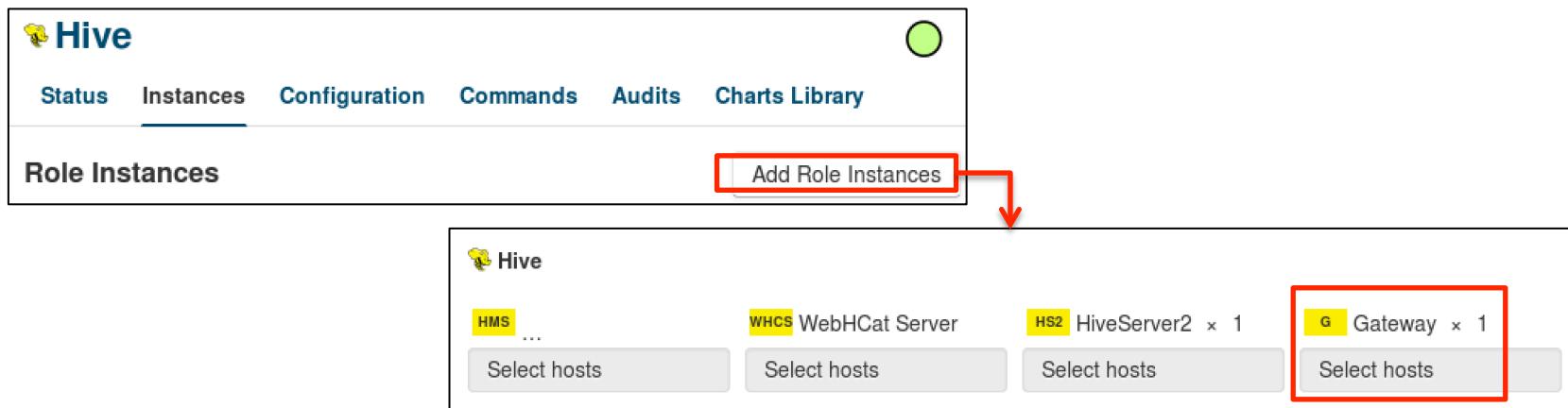
- What Are Hadoop Clients?
- **Installing and Configuring Hadoop Clients**
- Installing and Configuring Hue
- Hue Authentication and Authorization
- Hands-On Exercise: Using Hue to Control Hadoop User Access
- Conclusion

Installing Commonly-Used Hadoop Clients

Client	Installable from Cloudera Manager	Non-Cloudera Manager Installation Command (<code>sudo yum install</code>)
Hadoop Shell and MapReduce	✓	<code>hadoop-client</code>
Hive Shell	✓	<code>hive</code>
Pig Shell	Part of CDH parcel	<code>pig</code>
Sqoop Shell	✓	<code>sqoop</code>
Flume Agent	✓	<code>flume-ng-agent</code>
Centralized Hive Server	✓	<code>hive-server2</code>
Centralized Sqoop Server	✓	<code>sqoop2</code>
Oozie	✓	<code>oozie</code>

Installing Hadoop Clients on Hosts Managed by CM

- A **Gateway role** designates a host to receive client configurations for a service when the host does not have any other roles for the service on it
 - Enables Cloudera Manager to install and manage client configurations on that host
 - There is no process associated with a gateway role
 - Configure gateway roles for HDFS, Hive, Sqoop 1 Client, YARN and more
- Example – adding a Hive Gateway role



Installing Hadoop Clients on Hosts Not Managed by CM

- Hadoop client installers automatically include any required Hadoop APIs as dependencies if they are not already installed

```
$ sudo yum install hadoop-client
. .
=====
 Package           Arch      Version
=====
Installing:
hadoop-client          x86_64    2.5.0+cdh5.3.2+813-1.cdh5.3.2.p0.17.el6
Installing for dependencies:
avro-libs              noarch    1.7.6+cdh5.3.2+87-1.cdh5.3.2.p0.17.el6
bigtop-jsvc            x86_64    0.6.0+cdh5.3.2+621-1.cdh5.3.2.p0.17.el6
bigtop-utils           noarch    0.7.0+cdh5.3.2+0-1.cdh5.3.2.p0.17.el6
hadoop                 x86_64    2.5.0+cdh5.3.2+813-1.cdh5.3.2.p0.17.el6
hadoop-0.20-mapreduce  x86_64    2.5.0+cdh5.3.2+813-1.cdh5.3.2.p0.17.el6
hadoop-hdfs            x86_64    2.5.0+cdh5.3.2+813-1.cdh5.3.2.p0.17.el6
hadoop-mapreduce       x86_64    2.5.0+cdh5.3.2+813-1.cdh5.3.2.p0.17.el6
hadoop-yarn             x86_64    2.5.0+cdh5.3.2+813-1.cdh5.3.2.p0.17.el6
parquet                noarch    1.5.0+cdh5.3.2+62-1.cdh5.3.2.p0.17.el6
parquet-format          noarch    2.1.0+cdh5.3.2+8-1.cdh5.3.2.p0.17.el6
zookeeper              x86_64    3.4.5+cdh5.3.2+83-1.cdh5.3.2.p0.17.el6
. . .
```

Configuring Hadoop Clients

- **Cloudera Manager generates client configuration files with all needed configuration details to connect to Hadoop services**
- **For Clients managed by Cloudera Manager**
 - Configuration files are deployed to any host that is a client for a service
 - Hadoop daemons acting as Hadoop clients
 - Gateway role instances
 - Deployment is automatic when configuration changes are made *or* CM prompts to redeploy them via the web UI
- **For Clients not managed by Cloudera Manager**
 - Install the client software (without CM), then manually download the client configurations files from CM
 - Download client configurations for multiple cluster services or only specific services



Configuring Hadoop Clients on Hosts Not Managed by CM

1. From the Cloudera Manager Home page, from the menu for the cluster, choose “View Client Configuration URLs”
2. Download the client configurations for the given service

Client Configuration URLs		
Name	Type	URL
 YARN (MR2 Included)	YARN (MR2 Included)	 /cmf/services/5/client-config
 HDFS	HDFS	 /cmf/services/6/client-config
 Spark	Spark	 /cmf/services/8/client-config
 Sqoop 1 Client	Sqoop 1 Client	 /cmf/services/12/client-config
 Hive	Hive	 /cmf/services/16/client-config

3. Extract the client configuration files on the client
 - For example to /etc/hadoop/conf

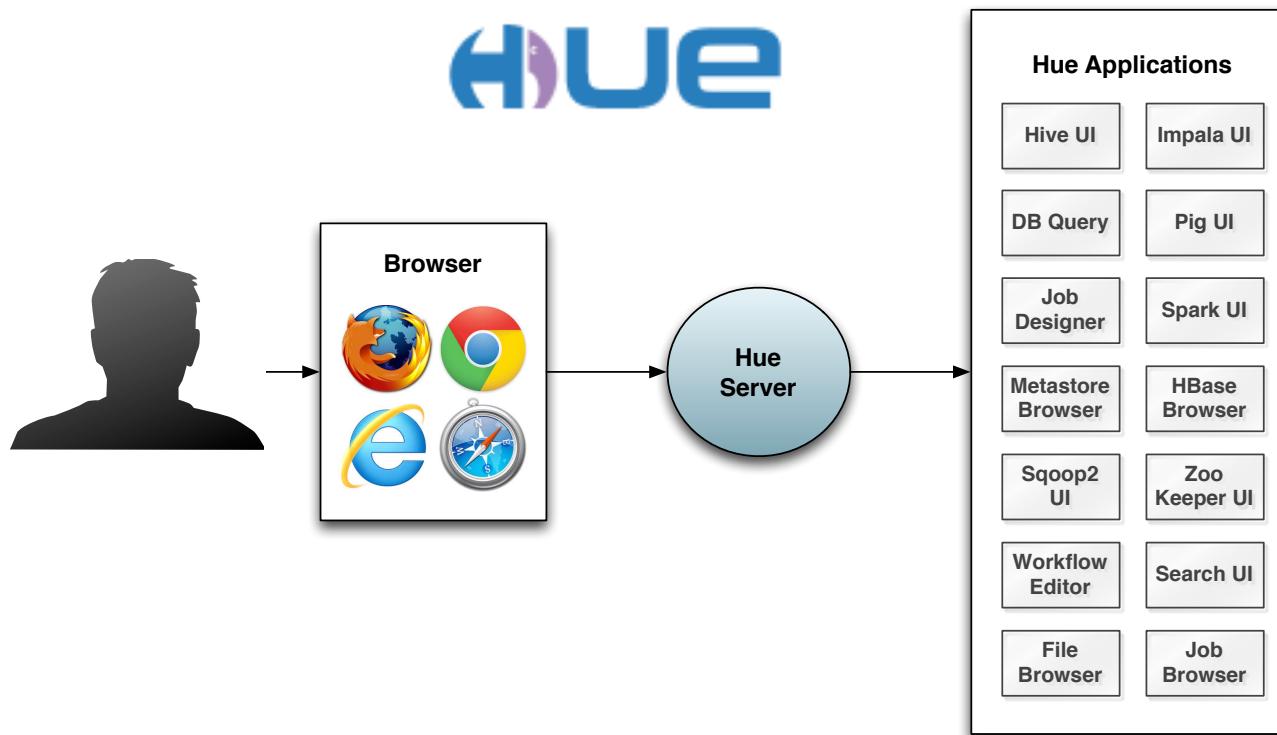
Chapter Topics

Hadoop Clients

- What Are Hadoop Clients?
- Installing and Configuring Hadoop Clients
- **Installing and Configuring Hue**
- Hue Authentication and Authorization
- Hands-On Exercise: Using Hue to Control Hadoop User Access
- Conclusion

What Is Hue?

- Hue provides a web interface for interacting with a Hadoop cluster
 - Hue Applications run in the browser (no client-side installation)



Installing, Starting, and Accessing Hue

■ Installing Hue

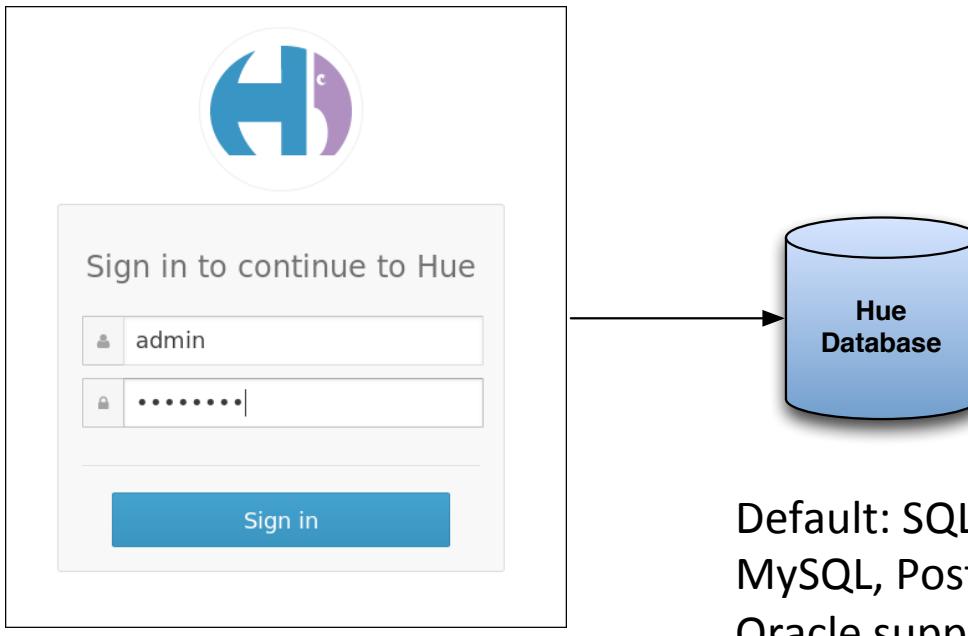
- Before installing Hue, verify services are installed on the cluster
 - Required components:
 - HDFS, Oozie, Hive
 - Optional components to enable more Hue applications:
 - HttpFS, Cloudera Search, Spark, YARN, Impala, HBase, Sentry, Pig, Sqoop, Sqoop 2, ZooKeeper
- In Cloudera Manager, choose ‘Add a Service’
 - Choose to install Hue

■ Start the Hue Server from the Hue Instances page in Cloudera Manager

■ Access Hue from a browser

- `http://<hue_server>:8888`

First User Login



- The first user to log in to Hue automatically receives superuser privileges
- Superusers can be added and removed
- The first user's login credentials are stored in the Hue database

Configuring Hue

- Log in to Hue as the superuser, then click on the ‘Configuration’ tab
 - `http://<hue_server>:8888/desktop/dump_config`

The screenshot shows the Hue configuration interface. At the top, there's a navigation bar with links like 'Query Editors', 'Data Browsers', 'Workflows', 'Search', 'Security', 'File Browser', 'Job Browser', and 'admin'. Below the navigation bar, the 'Configuration' tab is selected. A sidebar on the left lists 'Installed Applications' such as about, beeswax, desktop, filebrowser, hadoop, hbase, help, impala, indexer, jobbrowser, jobsub, libauth, liboozie, libopenid, librdbms, libsaml, libssentry, libsolr, metastore, oozie, pig, proxy, rdbms, search, security, sqoop, useradmin, and zookeeper. The main content area is titled 'Configuration Sections and Variables'. It shows a list of configuration items with their current values and descriptions. The items listed are:

- server_conn_timeout**: Value 120. Description: Timeout in seconds for Thrift calls. Default: 120.
- hive_conf_dir**: Value /var/run/cloudera-scm-agent/process/155-hue-HUE_SERVER/hive-conf. Description: Hive configuration directory, where hive-site.xml is located. Default: /var/run/cloudera-scm-agent/process/155-hue-HUE_SERVER/hive-conf.
- hive_server_port**: Value 10000. Description: Configure the port the HiveServer2 server runs on. Default: 10000.

Select Hue Applications - Requirements

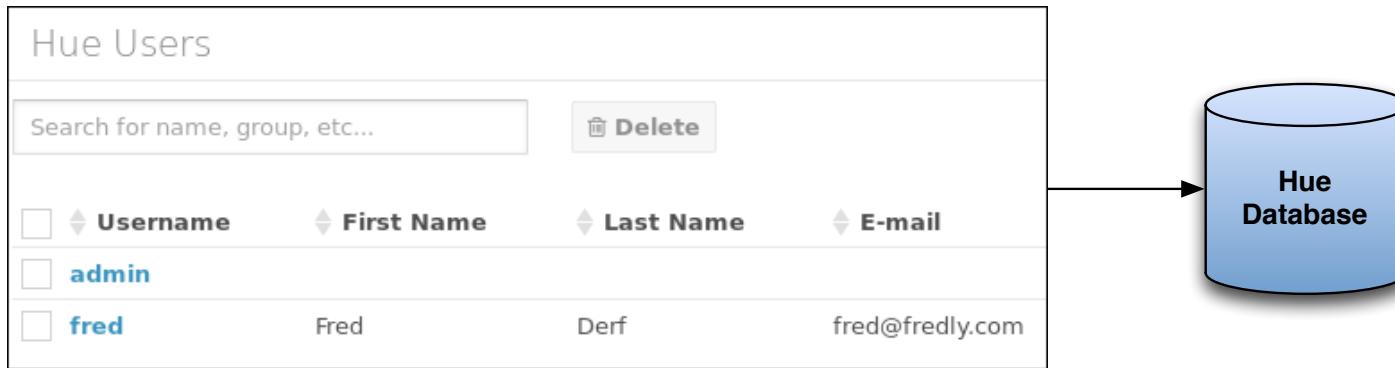
Hue Application	Required Component(s) and Configuration(s)
Hive Query Editor	<ul style="list-style-type: none">HiveServer2 installed
Impala Query Editor	<ul style="list-style-type: none">A shared Hive metastoreImpala service installed'Impala client configurations on Hue Server host machine
Pig Editor	<ul style="list-style-type: none">Pig installed on the machine running the Hue Server
Metastore Manager	<ul style="list-style-type: none">HiveServer2 installed
File Browser	<ul style="list-style-type: none">HttpFS installed or WebHDFS enabledNote that HttpFS is required for HDFS HA deployments
Job Browser	<ul style="list-style-type: none">YARN (MR2 Included) installedOptionally Spark installed

Chapter Topics

Hadoop Clients

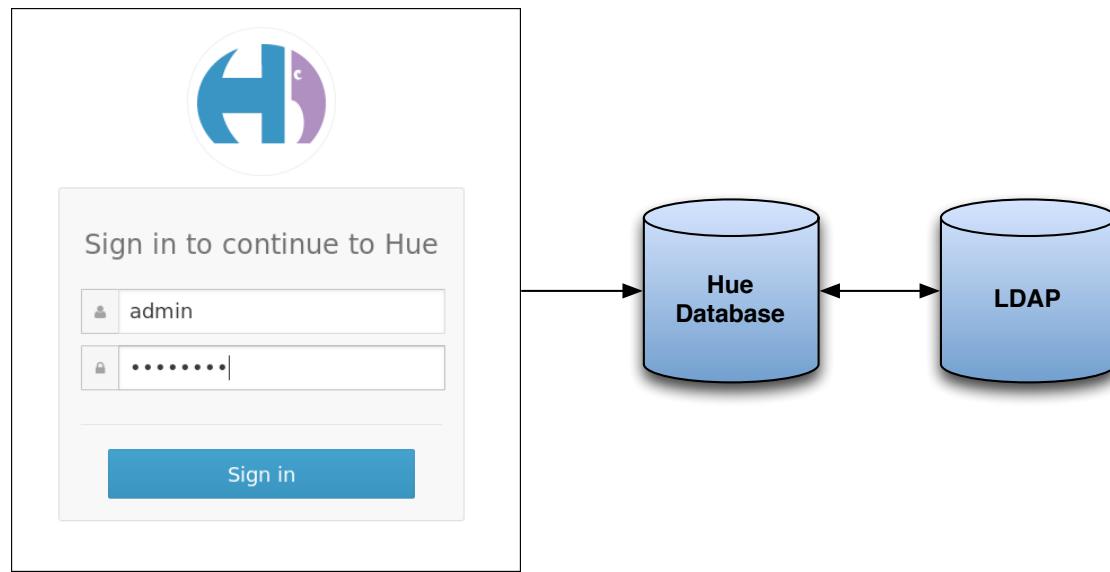
- What Are Hadoop Clients?
- Installing and Configuring Hadoop Clients
- Installing and Configuring Hue
- **Hue Authentication and Authorization**
- Hands-On Exercise: Using Hue to Control Hadoop User Access
- Conclusion

Defining Users and Groups With the Hue User Admin App



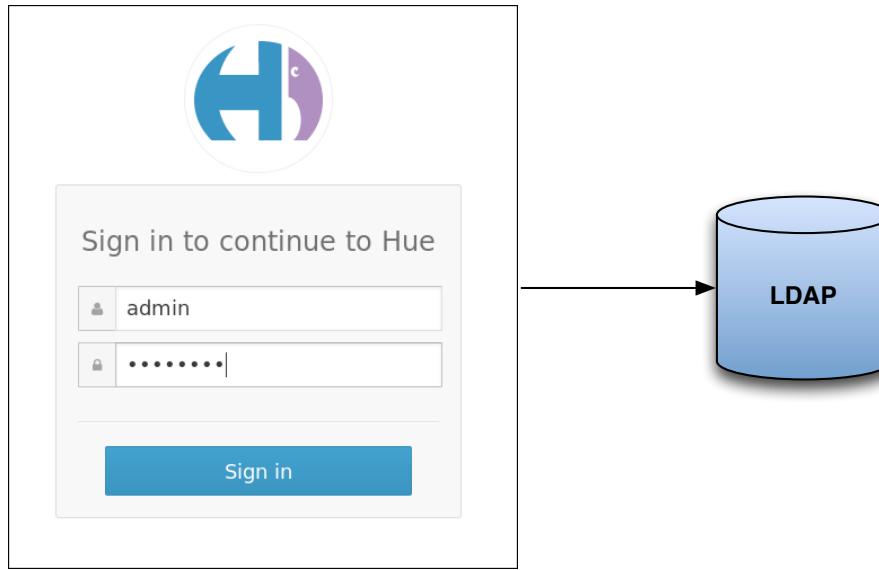
- Use Hue itself to maintain Hue users
- All user information, including credentials, is stored in the Hue Database

Accessing Users and Groups From LDAP – Option 1



- Administrator configures Hue to access the LDAP directory
- Administrator then syncs users and groups from LDAP to the Hue Database
- Hue authenticates users by accessing credentials from the Hue Database

Accessing Users and Groups From LDAP – Option 2



- Administrator configures Hue to access the LDAP directory
- Hue authenticates users by accessing credentials from LDAP

Restricting Access to Hue Applications

The screenshot shows a 'permissions' configuration screen. At the top is a checkbox labeled 'Select all'. Below it is a section for 'F' containing a checked checkbox for 'filebrowser.access:Launch this application(3)'. The section for 'H' contains three checkboxes: 'hbase.write:Allow writing in the HBase app.(4)', 'hbase.access:Launch this application(5)', and a checked checkbox for 'help.access:Launch this application(6)'. The section for 'I' contains a checked checkbox for 'impala.access:Launch this application(7)'.

Category	Permission	Description	Status
F	filebrowser.access:Launch this application	(3)	Selected
H	hbase.write:Allow writing in the HBase app.	(4)	Not Selected
H	hbase.access:Launch this application	(5)	Not Selected
H	help.access:Launch this application	(6)	Selected
I	impala.access:Launch this application	(7)	Selected

- Select a group in the Hue Manage Users application
- Edit the permissions for the group
 - Default permissions allow group members to access every Hue application

Chapter Topics

Hadoop Clients

- What Are Hadoop Clients?
- Installing and Configuring Hadoop Clients
- Installing and Configuring Hue
- Hue Authentication and Authorization
- **Conclusion**
- Hands-On Exercise: Using Hue to Control Hadoop User Access

Essential Points

- **Hadoop clients access functionality within a Hadoop cluster using the Hadoop API**
- **Hadoop clients require Hadoop configuration settings**
 - Cloudera Manager generates the needed configuration files
 - Client centralization using servers such as Hue, HiveServer2, and Sqoop2 reduces the need to distribute configuration settings to end users
- **Hue provides many capabilities to end users**
 - Browsing HDFS files
 - Designing, submitting, and browsing MapReduce jobs
 - Editing and submitting Hive and Impala queries
 - Designing, running, and viewing the status of Oozie workflows
- **Hue users are authenticated; access to functionality can be restricted**
 - Hue provides LDAP integration

Chapter Topics

Hadoop Clients

- What Are Hadoop Clients?
- Installing and Configuring Hadoop Clients
- Installing and Configuring Hue
- Hue Authentication and Authorization
- Conclusion
- **Hands-On Exercise: Using Hue to Control Hadoop User Access**

Hands-On Exercise: Using Hue to Control Hadoop User Access

- In this Hands-On Exercise, you will install Hue, test Hue applications, and then configure a limited-access Hadoop environment for analysts
- Please refer to the Hands-On Exercise Manual
 - Components deployed on your cluster after exercise completion:

	elephant	horse	monkey	tiger	lion
HDFS NameNode	✓				
HDFS SecondaryNameNode				✓	
HDFS DataNode	✓	✓	✓	✓	
HDFS Balancer		✓			
HDFS HttpFS			✓		
YARN (MR2 Included) NodeManager	✓	✓	✓	✓	
YARN (MR2 Included) ResourceManager		✓			
YARN (MR2 Included) JobHistory Server			✓		
ZooKeeper Server	✓	✓		✓	
Sqoop 1 Client Gateway	✓				
Flume Agent	✓		✓		
HiveServer2	✓				
Hive Gateway	✓				
Hive Metastore Server	✓				
Impala Catalog Server			✓		
Impala StateStore			✓		
Impala Daemon	✓	✓	✓	✓	
Spark Gateway	✓		✓		
Spark History Server			✓		
Hue Server			✓		
Oozie Server			✓		
Cloudera Manager Server					✓
Cloudera Manager Server Database					✓
Cloudera Management Services					✓
Cloudera Manager Agent	✓	✓	✓	✓	✓



Advanced Cluster Configuration

Chapter 11



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- **Advanced Cluster Configuration**
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Advanced Cluster Configuration

In this chapter, you will learn:

- **How to perform advanced configuration of Hadoop**
- **How to configure port numbers used by Hadoop**
- **How to configure HDFS rack awareness**
- **How to enable HDFS high availability**

Chapter Topics

Advanced Cluster Configuration

- **Advanced Configuration Parameters**
- Configuring Hadoop Ports
- Configuring HDFS for Rack Awareness
- Configuring HDFS High Availability
- Conclusion
- Hands-On Exercise: Configuring HDFS for High Availability

Advanced Configuration Parameters

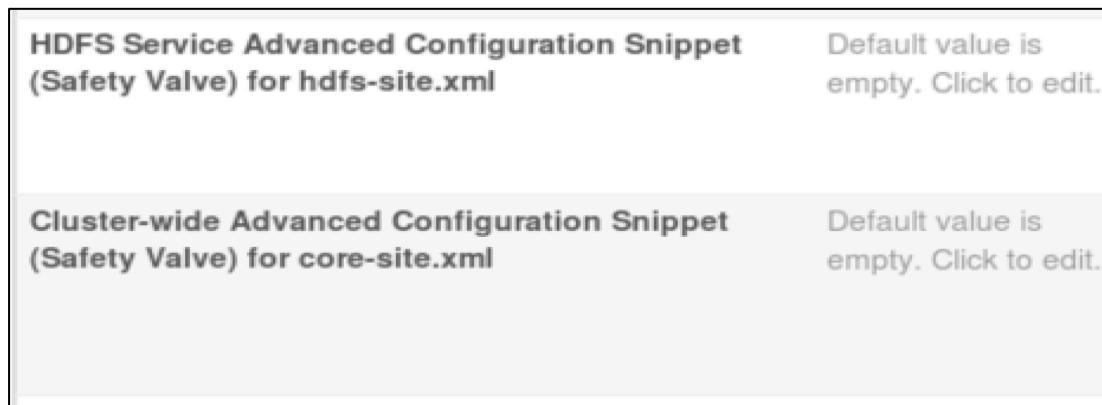
- **We have previously covered Hadoop's basic properties**
 - Essentially, the minimum needed to set up a working cluster
- **We will now discuss some additional properties**
- **These generally fall into one of several categories**
 - Optimization and performance tuning
 - Capacity management
 - Access control
- **The configuration recommendations in this section are baselines**
 - Use them as starting points, then adjust as required by the job mix in your environment

CDH Configuration Management (1)

- **Most common CDH settings are exposed in Cloudera Manager**
 - Can be set in the Configuration page for a service, role instance, or host
- **Cloudera Manager-generated CDH configurations**
 - Deployed to `/var/run/cloudera-scm.../process` subdirectories
- **Cloudera Manager does not expose *all* CDH configurations**
 - Reference for all CDH configurations:
<http://archive-primary.cloudera.com/cdh5/cdh/5/hadoop/>
 - See documentation under the ‘configuration’ menu at page bottom
 - When Cloudera Manager does not explicitly set a CDH property:
 - The default property value bundled with CDH will apply
 - These default settings are often stored in JAR files, for example in
`/opt/cloudera/parcels/CDH/lib/...`

Cloudera Manager – Advanced Configurations

- Use the “Advanced Configuration Snippet” safety-valve feature of Cloudera Manager to override not otherwise exposed CDH defaults
 - From the service’s configuration page, search for “Advanced Configuration Snippet” or “Safety Valve”
 - Add the settings to the appropriate Safety Valve in the format expected
 - *Example:* HDFS Configuration / Service-Wide / Advanced



- Properties can be set cluster-wide or for a specific role group or role
 - Configurations are for server or client configurations

Aside: The Cloudera Manager REST API

- **It is possible to programmatically manage Hadoop clusters using the Cloudera Manager REST API**
 - Extensive functionality:
 - Deploy a cluster, configure services, start, stop, failover services, configure HA, monitor services, hosts, and jobs, and more
 - Import/export entire cluster configuration
 - Available with Cloudera Express and Cloudera Enterprise
 - Includes open source client libraries
- **Access at `http://cm-host:7180/api/v9/`**
 - Available as soon as Cloudera Manager is installed
 - Easy access via `curl`
 - Python or Java clients recommended for serious use
- **The API accepts HTTP POST, GET, PUT, and DELETE methods**
 - Accepts and returns JSON formatted data

HDFS – NameNode Tuning

dfs.namenode.handler.count

Set in HDFS / NameNode Group / Performance

- The number of server threads for the NameNode
- Threads used for RPC calls from clients and DataNodes (heartbeats and metadata operations)
- Default in Cloudera Manager: 30 (non-CM default is 10)
- Recommended: Natural logarithm of the number of cluster nodes x 20
- Symptoms of this being set too low: ‘connection refused’ messages in DataNode logs as they try to transmit block reports to the NameNode
- Used by the NameNode

HDFS – DataNode Tuning

dfs.datanode.failed.volumes.tolerated

Set in HDFS / DataNode Group

- The number of volumes allowed to fail before the DataNode takes itself offline, ultimately resulting in all of its blocks being re-replicated
- CM Default: 0
- Often increased on machines with several disks
- Used by DataNodes

dfs.datanode.max.locked.memory

Set in HDFS / DataNode Group / Resource Management

- The maximum amount of memory (in bytes) a DataNode can use for caching
- CM Default: 4GB
- Must be less than the value of the OS configuration property `ulimit -l` for the DataNode user
- Used by DataNodes

File Compression (1)

io.compression.codecs

Set in HDFS / Service-Wide

- List of compression codecs that Hadoop can use for file compression
- If you are using another codec, add it here
- CM default value includes the following org.apache.hadoop.io.compress codecs: DefaultCodec, GzipCodec, BZip2Codec, DeflateCodec, SnappyCodec, Lz4Codec
- Used by clients and all nodes running Hadoop daemons

MapReduce – Reducer Scheduling and Input Fetch

mapreduce.job.reduce.slowstart.completedmaps

Set in YARN / Gateway Group

- The percentage of Map tasks which must be completed before the ResourceManager will schedule Reducers on the cluster
- CM Default: 0.8 (80 percent)
- Recommendation: 0.8 (80 percent)
- Used by the ResourceManager

mapreduce.reduce.shuffle.parallelcopies

Set in YARN / Gateway Group

- Number of threads a Reducer can use in parallel to fetch Mapper output
- CM Default: 10
- Recommendation: $\ln(\text{number of cluster nodes}) * 4$ with a floor of 10
- Used by ShuffleHandler

MapReduce – Speculative Execution

- If a MapReduce task is running significantly more slowly than the average speed of tasks for that job, speculative execution may occur
 - Another attempt to run the same task is instantiated on a different node
 - Results from the first completed task are used, the slower task is killed

mapreduce.map.speculative

Set in YARN / Gateway Group

- Whether to allow speculative execution for Map tasks
- CM Default: false, Recommendation: false
- Used by MapReduce ApplicationMasters

mapreduce.reduce.speculative

Set in YARN / Gateway Group

- Whether to allow speculative execution for Reduce tasks
- CM Default: false, Recommendation: false
- Used by MapReduce ApplicationMasters

Chapter Topics

Advanced Cluster Configuration

- Advanced Configuration Parameters
- **Configuring Hadoop Ports**
- Configuring HDFS for Rack Awareness
- Configuring HDFS High Availability
- Conclusion
- Hands-On Exercise: Configuring HDFS for High Availability

Common Hadoop Ports

- **Hadoop daemons each provide a Web-based user interface**
 - Useful for both users and system administrators
- **Expose information on a variety of different ports**
 - Port numbers are configurable, although there are defaults for most
- **Hadoop also uses various ports for components of the system to communicate with each other**
- **Full list of ports used by components of CDH 5:**
 - http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cdh_ig_ports_cdh5.html
- **All ports used in a Cluster are listed in one location in Cloudera Manager**
 - From the Cluster page's Configuration menu, choose "All Port Configurations"

Web UI Ports for Users

	Daemon	Default Port	Configuration parameter
HDFS	NameNode	50070	dfs.namenode.http-address
	DataNode	50075	dfs.datanode.http.address
	Secondary NameNode	50090	dfs.namenode.secondary.http-address
YARN	ResourceManager	8088	yarn.resourcemanager.webapp.address
	NodeManager	8042	yarn.nodemanager.webapp.address
	MR JobHistoryServer	19888	mapreduce.jobhistory.webapp.address
Other	Cloudera Manager	7180	Configure in Cloudera Manager by going to Administration > Settings > Ports and Addresses
	Hue	8888	http_port
	Spark History Server	18088	history.port

Hadoop Ports for Administrators (1)

Daemon	Default Port	Configuration Parameter	Used for
NameNode	8020	fs.defaultFS	Filesystem metadata operations
DataNode	50010	dfs.datanode.address	DFS data transfer
	50020	dfs.datanode.ipc.address	Block metadata operations and recovery
Resource Manager	8032	yarn.resourcemanager.address	Application submission; used by clients
	8033	yarn.resourcemanager.admin.address	Administration RPC server port; used by the <code>yarn rmadmin</code> client
	8030	yarn.resourcemanager.scheduler.address	Scheduler RPC port; used by ApplicationMasters
	8031	yarn.resourcemanager.resource-tracker.address	Resource tracker RPC port; used by NodeManagers

Hadoop Ports for Administrators (2)

Daemon	Default Port	Configuration Parameter	Used for
NodeManager	8040	yarn.nodemanager.localizer.address	Resource localization RPC port
JobHistoryServer	10020	mapreduce.jobhistory.address	JobHistoryServer RPC port; used by clients to query job history.
ShuffleHandler (MapReduce's auxiliary service)	13562	mapreduce.shuffle.port	ShuffleHandler's HTTP port; used for serving Mapper outputs

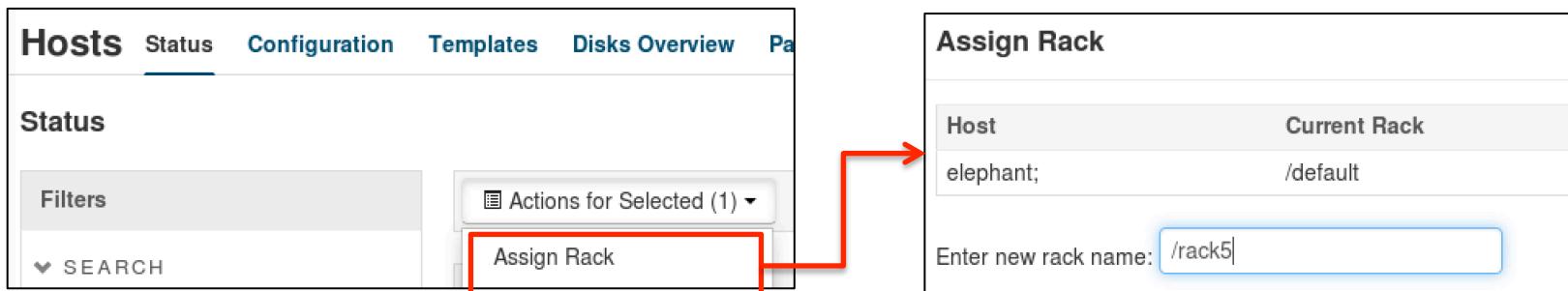
Chapter Topics

Advanced Cluster Configuration

- Advanced Configuration Parameters
- Configuring Hadoop Ports
- **Configuring HDFS for Rack Awareness**
- Configuring HDFS High Availability
- Conclusion
- Hands-On Exercise: Configuring HDFS for High Availability

HDFS – Rack Topology Awareness

- Recall that HDFS is ‘rack aware’
 - Distributes blocks based on hosts’ locations
- To maximize performance, specify network locations of hosts and racks
 - Important for clusters that span more than one rack
 - In the Hosts page in Cloudera Manager, assign hosts to racks
 - Specify Rack ID in the form /datacenter/rack



- Any host without a specified rack location shows /default location

Chapter Topics

Advanced Cluster Configuration

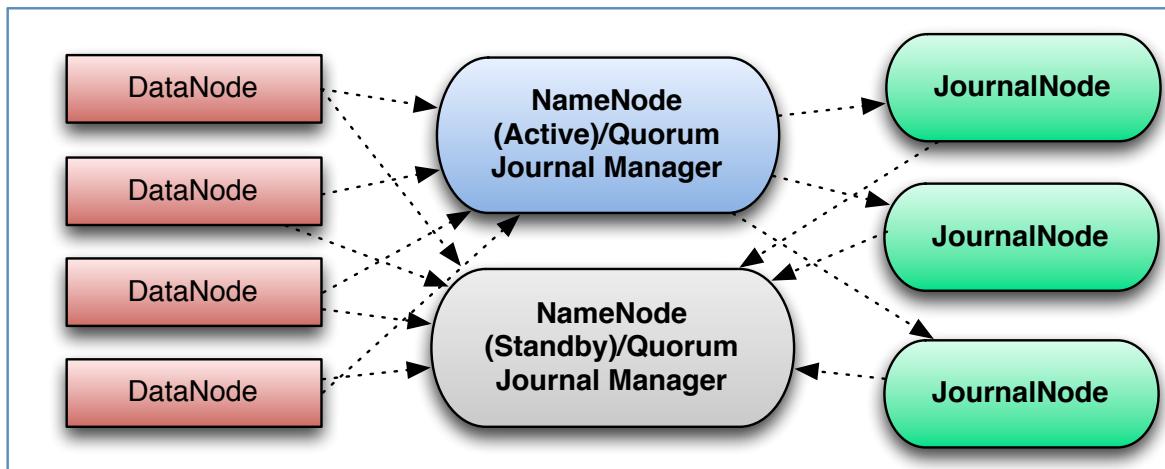
- Advanced Configuration Parameters
- Configuring Hadoop Ports
- Configuring HDFS for Rack Awareness
- **Configuring HDFS High Availability**
- Conclusion
- Hands-On Exercise: Configuring HDFS for High Availability

HDFS High Availability Overview

- A single NameNode is a single point of failure
- Two ways a NameNode can result in HDFS downtime
 - Unexpected NameNode crash (rare)
 - Planned maintenance of NameNode (more common)
- HDFS High Availability (HA) eliminates this SPOF
 - Available since CDH4 (or related Apache Hadoop 0.23.x, and 2.x)
- New Daemons that HDFS HA introduces to the cluster
 - NameNode (active)
 - NameNode (standby)
 - Failover Controllers
 - Journal Nodes
- The Secondary NameNode is not used in an HDFS HA configuration

HDFS High Availability Architecture (1)

- HDFS High Availability uses a pair of NameNodes
 - One Active and one Standby
 - Clients only contact the Active NameNode
 - DataNodes heartbeat in to both NameNodes
 - Active NameNode writes its metadata to a quorum of JournalNodes
 - Standby NameNode reads from the JournalNodes to remain in sync with the Active NameNode



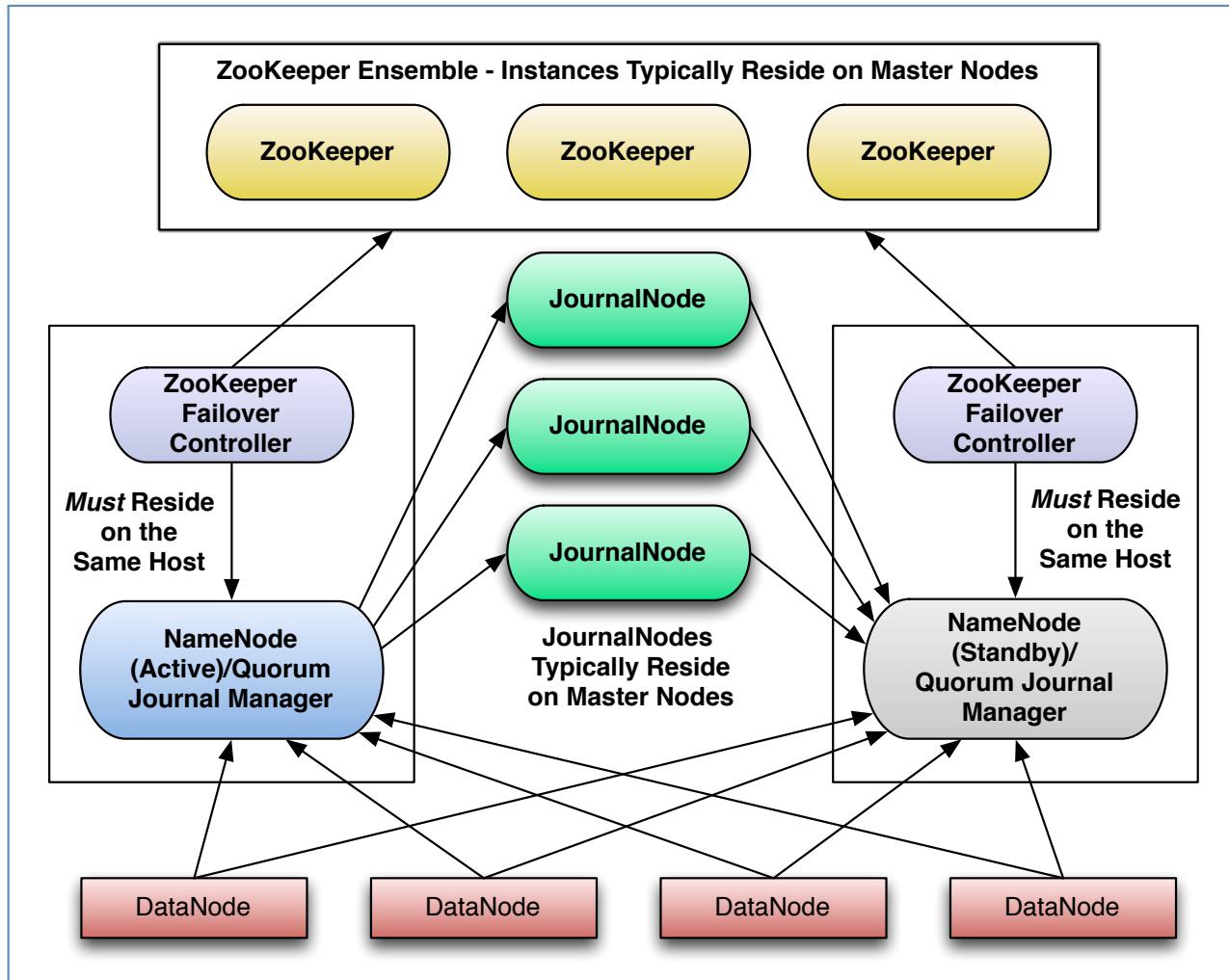
HDFS High Availability Architecture (2)

- **Active NameNode writes edits to the JournalNodes**
 - Software to do this is the *Quorum Journal Manager (QJM)*
 - Built in to the NameNode
 - Waits for a success acknowledgment from the majority of JournalNodes
 - Majority commit means a single crashed or lagging JournalNode will not impact NameNode latency
 - Uses the *Paxos* algorithm to ensure reliability even if edits are being written as a JournalNode fails
- **Note that there is no Secondary NameNode when implementing HDFS High Availability**
 - The Standby NameNode periodically performs checkpointing

Failover

- **Only one NameNode must be active at any given time**
 - The other is in standby mode
- **The standby maintains a copy of the active NameNode's state**
 - So it can take over when the active NameNode goes down
- **Two types of failover**
 - Manual (detected and initiated by a user)
 - Automatic (detected and initiated by HDFS itself)
 - Automatic failover uses ZooKeeper
 - When HA is configured, a daemon called the ZooKeeper Failover Controller (ZKFC) runs on each NameNode machine

HDFS HA With Automatic Failover – Deployment



HDFS High Availability and Automatic Failover

- **Cloudera Manager configures HA using Quorum-based storage**
 - Uses a quorum of JournalNodes
 - Each JournalNode maintains a local edits directory
 - That directory contains files detailing namespace metadata modifications
- **With HA configured, automatic failover is also available**
 - Cloudera Manager sets `dfs.ha.automatic-failover.enabled` to true for the NameNode role instances it configures for HA

Fencing

- **It is essential that exactly one NameNode be active**
 - Possibility of “split-brain syndrome” otherwise
- **The Quorum Journal Manager ensures that a previously-active NameNode cannot corrupt the NameNode metadata**
- **When configuring HDFS HA, one or more fencing methods must be specified**
 - Methods available:
 - shell – the Cloudera Manager default (uses CM Agent). Configure in HDFS / Configuration / Service-Wide / High Availability
 - sshfence – connects to the target node via ssh and kills the process listening on service’s TCP port
 - In order for failover to occur, one of the methods must run successfully

HDFS HA Deployment - Without Cloudera Manager

- **Without Cloudera Manager, enabling High Availability for HDFS is a long, complex, error prone process**
 - Overview of major steps involved
 1. Modify multiple Hadoop configurations across all DataNodes
 2. Install and start the JournalNodes
 3. If not already installed, configure and start a ZooKeeper ensemble
 4. Initialize the shared edits directory if converting from a non-HA deployment
 5. Install, bootstrap, and start the Standby NameNode
 6. Install, format, and start the ZooKeeper failover controllers
 7. Restart DataNodes and the YARN and MapReduce daemons
 - **With Cloudera Manager it is a simple process**

HDFS HA Deployment - With Cloudera Manager

- **Cloudera Manager makes it easy to enable HDFS HA**
 - Completes many complex steps to enable High Availability
 - Wizard guides you through the process

The screenshot shows the Cloudera Manager interface for the HDFS-1 cluster. The top navigation bar includes Home, Clusters (selected), Hosts, Diagnostics, Audits, Charts, Backup, and Administration. Below the navigation is a breadcrumb trail for Cluster 1. The main content area shows the HDFS-1 service with tabs for Status, Instances (selected), Configuration, Commands, Audits, File Browser, and Charts. On the right, there are Actions, Role Instances (with Add Role Instances and Role Groups buttons), and an 'Enable High Availability' button, which is highlighted with a red box. Other buttons include Roll Edits.

The screenshot shows the 'Enable High Availability for HDFS-1' wizard. The title is 'Enable High Availability for HDFS-1'. Under 'Getting Started', it says: 'This wizard leads you through adding a standby NameNode, restarting this HDFS service and any dependent services, and then re-deploying client configurations.' A 'Nameservice Name' field contains 'nameservice1'. Below it, a note states: 'Enabling High Availability creates a new nameservice. Accept the default name nameservice1 or provide another name in Nameservice Name.' A red arrow points from the 'Enable High Availability' button on the previous screen to this wizard page.

Enabling HDFS HA Using Cloudera Manager

- Three steps to Enable HDFS HA in Cloudera Manager
 - 1. Configure directories for JournalNodes to store edits directories
 - Set permissions on these new directories for the hdfs user
 - 2. Ensure the ZooKeeper service is installed and enabled for HDFS
 - 3. From the HDFS Instances page, run the "Enable High Availability" wizard
 - Specify the hosts for the two NameNodes and the JournalNodes
 - Specify the JournalNode Edits directory for each host
 - The wizard performs the necessary steps to enable HA
 - Including the creation and configuration of new Hadoop daemons

After Enabling HDFS HA

- **After enabling HDFS HA, some manual configurations may be needed**
 - For Hive
 - Upgrade the Hive Metastore
 - Consult the Cloudera documentation for details
 - For Impala
 - Complete the same steps as for Hive (above)
 - Next, run INVALIDATE_METADATA command from the Impala shell
 - For Hue
 - Add the HttpFS role (if not already on the cluster)

Chapter Topics

Advanced Cluster Configuration

- Advanced Configuration Parameters
- Configuring Hadoop Ports
- Configuring HDFS for Rack Awareness
- Configuring HDFS High Availability
- **Conclusion**
- Hands-On Exercise: Configuring HDFS for High Availability

Essential Points

- ‘Advanced Configuration Snippets’ provide a way to configure CDH parameters not otherwise exposed in Cloudera Manager
- Hadoop provides a rack awareness capability, which should be implemented to increase data locality
- HDFS can be configured for high availability with an automatic failover capability

Chapter Topics

Advanced Cluster Configuration

- Advanced Configuration Parameters
- Configuring Hadoop Ports
- Configuring HDFS for Rack Awareness
- Configuring HDFS High Availability
- Conclusion
- **Hands-On Exercise: Configuring HDFS for High Availability**

Hands-On Exercise: Configuring HDFS for High Availability

- In this Hands-On Exercise, you will configure your Hadoop cluster for HDFS high availability and then test for automatic NameNode failover
- Please refer to the Hands-On Exercise Manual
 - Cluster deployment after exercise completion (only a subset of the daemons on the cluster are shown):

	elephant	horse	monkey	tiger
HDFS NameNode	✓			
HDFS Secondary NameNode				removed
HDFS Standby NameNode				✓
JournalNode	✓	✓		✓
Failover Controller	✓			✓
HDFS DataNode	✓	✓	✓	✓
HDFS Balancer		✓		
HDFS HttpFS			✓	
YARN (MR2 Included) NodeManager	✓	✓	✓	✓
YARN (MR2 Included) ResourceManager		✓		
YARN (MR2 Included) JobHistory Server			✓	
ZooKeeper Server	✓	✓		✓



Hadoop Security

Chapter 12



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- **Hadoop Security**
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Hadoop Security

In this chapter you will learn:

- **Why security is important for Hadoop**
- **What Kerberos is and how it relates to Hadoop**
- **The Role of Sentry in Securing Hive and Impala**
- **The types of encryption available in CDH**
- **What to consider when securing Hadoop**

Chapter Topics

Hadoop Security

- **Why Hadoop Security Is Important**
- Hadoop's Security System Concepts
- What Kerberos Is and How it Works
- Securing a Hadoop Cluster with Kerberos
- Other Security Topics
- Conclusion

Why Hadoop Security Is Important

- **Laws governing data privacy**
 - Particularly important for healthcare and finance industries
- **Export control regulations for defense information**
- **Protection of proprietary research data**
- **Company policies**
 - Different teams in a company have different needs
- **Setting up multiple clusters is a common solution**
 - One cluster may contain protected data, another cluster does not

Chapter Topics

Hadoop Security

- Why Hadoop Security Is Important
- **Hadoop's Security System Concepts**
- What Kerberos Is and How it Works
- Securing a Hadoop Cluster with Kerberos
- Other Security Topics
- Conclusion

Defining Important Security Terms

■ Security

- Computer security is a very broad topic
- Access control and encryption are the areas most relevant to Hadoop
- We'll therefore focus on authentication, authorization, and encryption

■ Authentication

- Confirming the identity of a participant
- Typically done by checking credentials (username/password)

■ Authorization

- Determining whether a participant is allowed to perform an action
- Typically done by checking an access control list

■ Encryption

- Ensure only authorized users can access a data set
- OS filesystem-level, HDFS-level, and Network-level options

Types of Hadoop Security

- **HDFS file ownership and permissions**
 - Provides modest protection, but user/group authentication is easily subverted (client-side)
 - Mainly intended to guard against accidental deletions/overwrites
- **Enhanced security with Kerberos**
 - Optional – provides strong authentication of both clients and servers
 - Wraps Hadoop API calls in an SASL handshake
 - Applications can be run under the submitter's own account
- **Encrypted HDFS data transfers**
- **HDFS data at rest encryption**
- **Encrypted HTTP traffic**
 - The Hadoop Web UIs
 - Intermediate data transferred during shuffle and sort

Hadoop Security Design Considerations

- **The security of a cluster is enhanced by isolation**
 - It should ideally be on its own network
 - Access to nodes/network should be limited for untrusted users

Chapter Topics

Hadoop Security

- Why Hadoop Security Is Important
- Hadoop's Security System Concepts
- **What Kerberos Is and How it Works**
- Securing a Hadoop Cluster with Kerberos
- Other Security Topics
- Conclusion

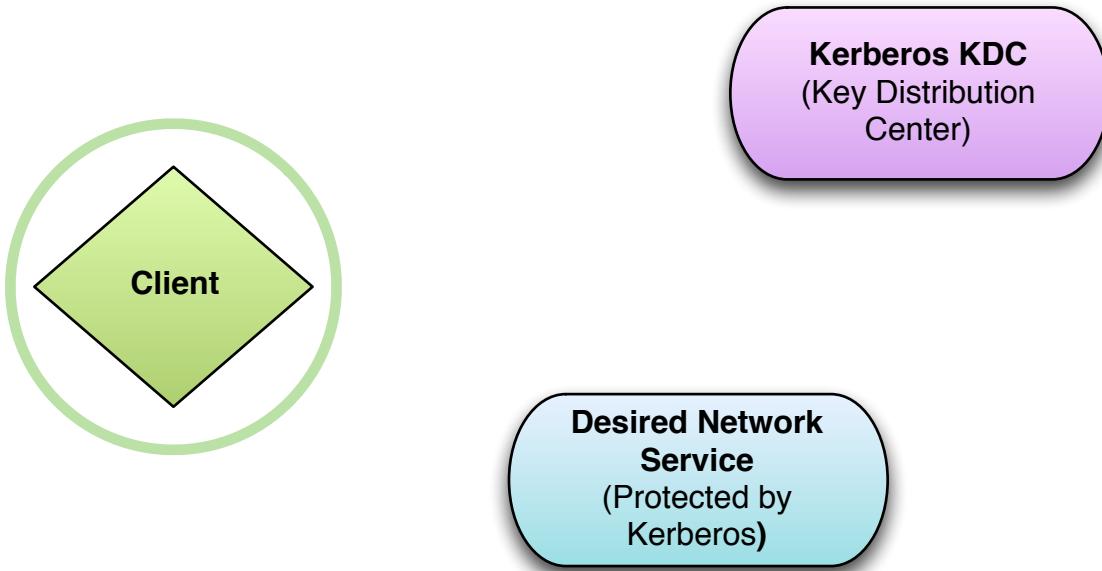
The Role of Kerberos in CDH5

- Hadoop daemons leverage Kerberos to perform user authentication on all remote procedure calls (RPCs)
- Group resolution is performed on master nodes to guarantee group membership is not manipulated

Kerberos Exchange Participants (1)

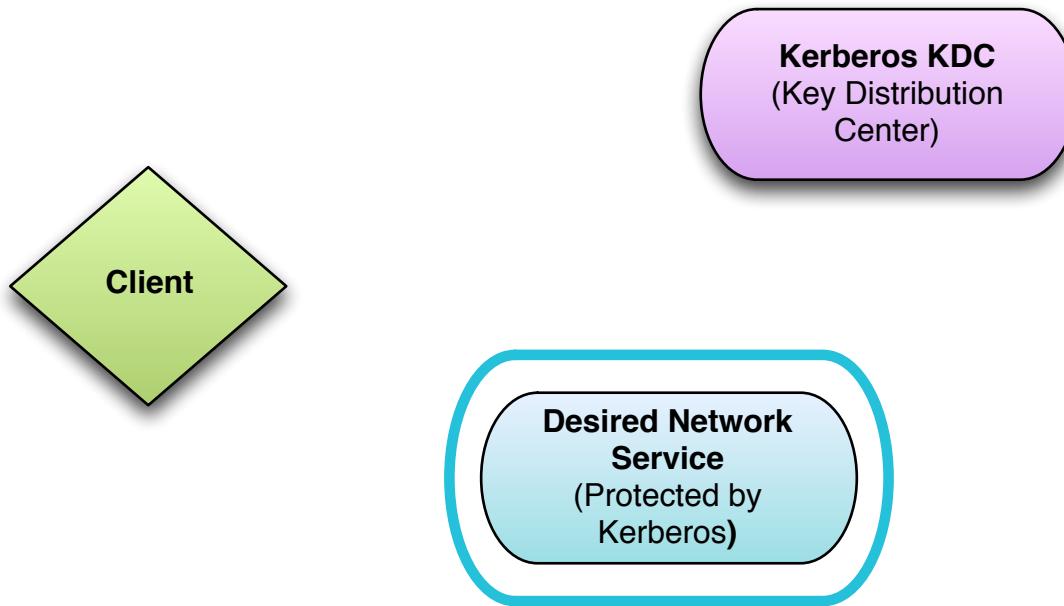
- Kerberos involves messages exchanged among three parties
 - The client
 - The server providing a desired network service
 - The Kerberos Key Distribution Center (KDC)

Kerberos Exchange Participants (2)



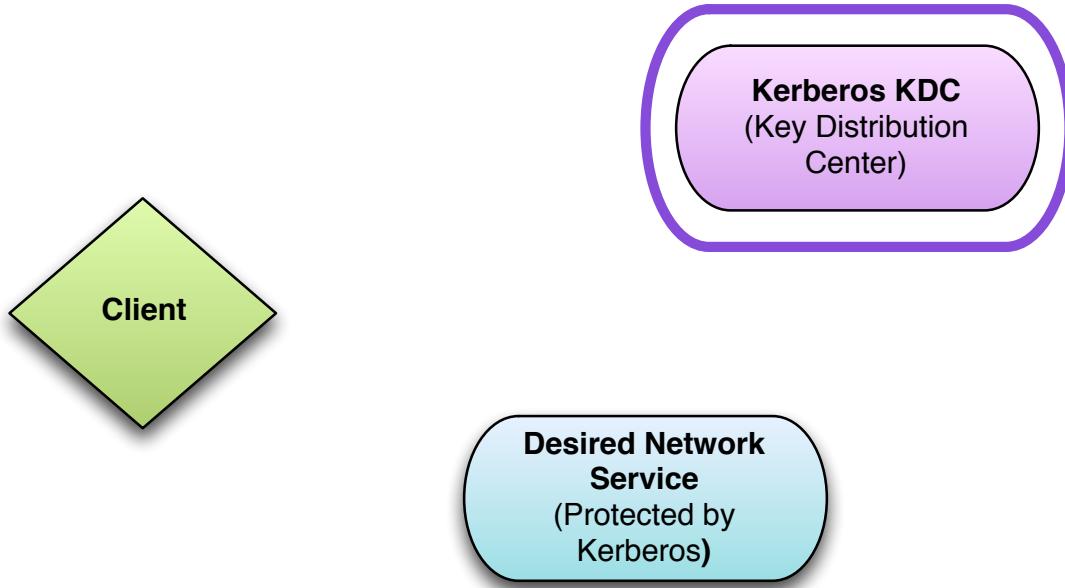
- **The client is software that desires access to a Hadoop service**
 - The `hdfs dfs` command is one example of a client

Kerberos Exchange Participants (3)



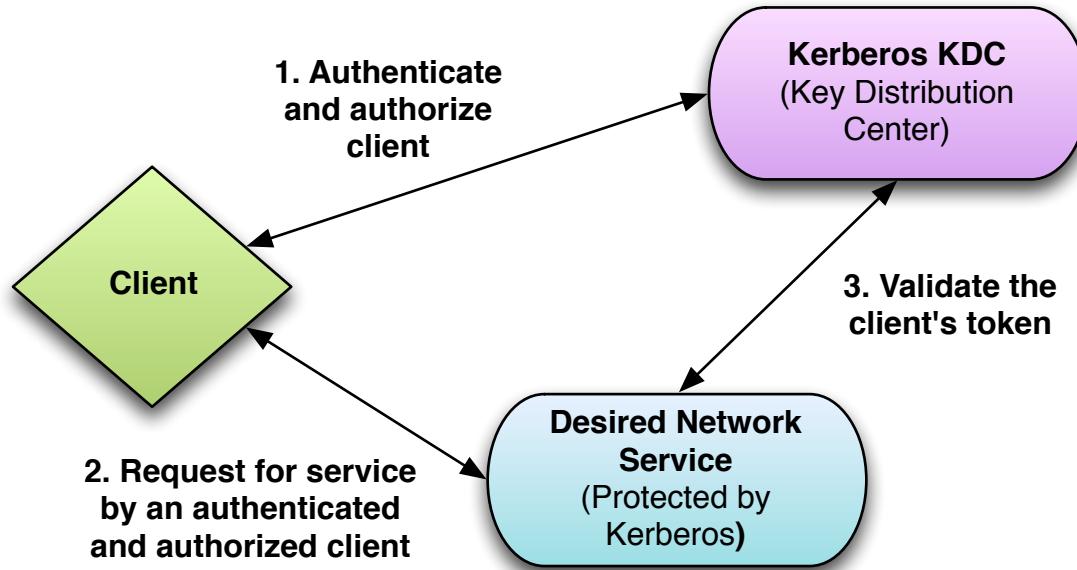
- **This is the Hadoop service the client wishes to access**
 - For Hadoop, this will be a service daemon (such as the NameNode)

Kerberos Exchange Participants (4)



- **The Kerberos server (KDC) authenticates and authorizes a client**
- **The KDC is neither part of nor provided by Hadoop**
 - Most Linux distributions come with the MIT Kerberos KDC
 - Microsoft Active Directory also provides a KDC

General Kerberos Concepts (1)



- **Kerberos is a standard network security protocol**
 - Currently at version 5 (RFC 4120)
 - Services protected by Kerberos don't directly authenticate the client

General Kerberos Concepts (2)

- **Authenticated status is cached**
 - You do not need to explicitly submit credentials with each request
- **Passwords are not sent across network**
 - Instead, passwords are used to compute encryption keys
 - The Kerberos protocol uses encryption extensively
- **Timestamps are an essential part of Kerberos**
 - Make sure you synchronize system clocks (NTP)
- **It is important that reverse lookups work correctly**

Kerberos Terminology

- **Knowing a few terms will help you with the documentation**
- **Realm**
 - A group of machines participating in a Kerberos network
 - Identified by an uppercase domain (EXAMPLE.COM)
- **Principal**
 - A unique identity which can be authenticated by Kerberos
 - Can identify either a host or an individual user
 - Every user in a secure cluster will have a Kerberos principal
- **Keytab file**
 - A file that stores Kerberos principals and associated keys
 - Allows non-interactive access to services protected by Kerberos

Chapter Topics

Hadoop Security

- Why Hadoop Security Is Important
- Hadoop's Security System Concepts
- What Kerberos Is and How it Works
- **Securing a Hadoop Cluster with Kerberos**
- Other Security Topics
- Conclusion

Hadoop Security Setup Prerequisites

- **Working Hadoop cluster**
 - Installing CDH from parcels or packages is strongly advised!
 - Ensure your cluster actually works before trying to secure it!
- **Working Kerberos KDC server**
- **Kerberos client libraries installed on all Hadoop nodes**

Configuring Hadoop Security (1)

- **Hadoop security configuration is a specialized topic**
- **Many specifics depend on**
 - Version of Hadoop and related programs
 - Type of Kerberos server used (Active Directory or MIT)
 - Operating system and distribution
- **You must follow instructions exactly**
 - There is little room for misconfiguration
 - Mistakes often result in vague “access denied” errors
 - May need to work around version-specific bugs

Configuring Hadoop Security (2)

- **For these reasons, we can't cover this in depth during class**
- **See the “CDH Security Guide” for detailed instructions**
 - Available at <http://www.cloudera.com/content/cloudera/en/documentation/cdh5/v5-0-0/CDH5-Security-Guide/CDH5-Security-Guide.html>
 - Be sure to read the guide corresponding to your version of CDH
- **The CDH Security Guide documents manual configuration**
 - Configuring Hadoop with Kerberos involves many tedious steps
 - Cloudera Manager (Enterprise) automates many of them

Securing Related Services

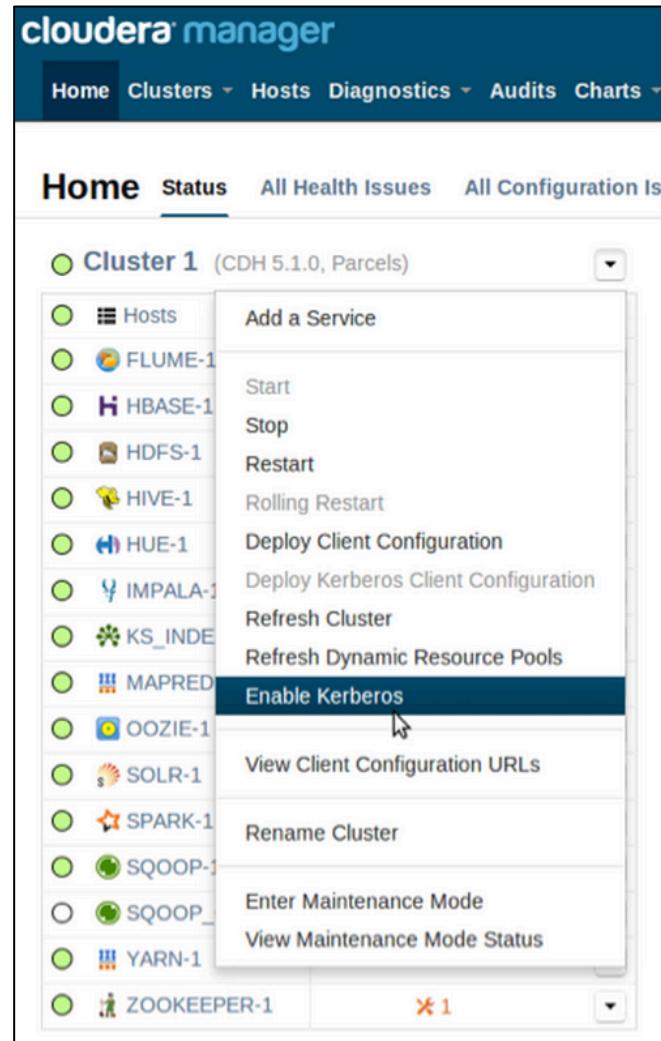
- There are many “ecosystem” tools that interact with Hadoop
- Most require minor configuration changes for a secure cluster
 - Such as specifying Kerberos principals or keytab file paths
- Exact configuration details vary with each tool
 - See documentation for details
- Some require no configuration changes at all
 - Such as Pig and Sqoop

Active Directory Integration

- **Microsoft's Active Directory (AD) is a enterprise directory service**
 - Used to manage user accounts for a Microsoft Windows network
- **Recall that every Hadoop user must have a Kerberos principal**
 - It can be tedious to set up all these accounts
 - Many organizations would prefer to use AD for Hadoop users
- **Cloudera's recommended approach**
 - Integrate with Active Directory KDC or run a local MIT Kerberos KDC
 - Create all service principals (like `hdfs` and `mapred`) in this realm
- **Instructions can be found in Cloudera's CDH Security Guide**

Direct Active Directory Integration with Kerberos

- Integrate with an existing Active Directory KDC
- Wizard in Cloudera Manager enables Kerberos for the cluster
 - Generates and deploys Kerberos client configuration (krb5.conf)
 - Configures CDH components
 - Generates principals needed by all processes running in the cluster



Chapter Topics

Hadoop Security

- Why Hadoop Security Is Important
- Hadoop's Security System Concepts
- What Kerberos Is and How it Works
- Securing a Hadoop Cluster with Kerberos
- **Other Security Topics**
- Conclusion

Encryption Overview

- **Encryption seeks to ensure that only authorized users can view, use, or contribute to a data set**
- **Data protection can be applied at three levels within Hadoop:**
 - Linux filesystem level
 - Example: Cloudera Navigator Encrypt
 - HDFS level
 - Encryption applied by the HDFS client software
 - HDFS Data At Rest Encryption operates at the HDFS folder level, enabling encryption to be applied only to the HDFS folders where it is needed. Navigator Key Trustee should be used
 - Network level
 - Encrypt data just before sending across a network and decrypt it as it is received. This protection uses industry-standard protocols such as SSL/TLS

OS Filesystem Level Encryption – Navigator Encrypt

- **Production ready, and included with the Cloudera Navigator license**
- **Operations at the Linux volume level**
 - Capable of encrypting cluster data inside and outside of HDFS
 - No change to application code required
- **Provides a transparent layer between the application and file system reducing the performance impact of encryption**
- **Navigator Key Trustee**
 - The default HDFS encryption uses a local Java keystore
 - Navigator Key Trustee is a ‘virtual safe-deposit box’ keystore server for managing encryption keys, certificates, and passwords
 - Encryption keys stored separately from encrypted data

HDFS Level “Data at Rest” Encryption

- **Provides transparent end-to-end encryption of data read from and written to HDFS**
 - No change to application code required
 - Data encrypted and decrypted only by the HDFS client
 - HDFS does not store or have access to unencrypted data or keys
- **Production ready with the CM 5.3 release**
- **Operates at the HDFS folder level**
 - Apply to folders where needed
- **Deployment Overview**
 - Use Cloudera Manager to deploy the Hadoop Key Management Server (KMS) service for storing keys
 - Create Encryption Zones, then add files to the Encryption Zones

Network Level Encryption

- **Transport Layer Security (TLS)**
 - Provides communication security over the network to prevent snooping
- **Enable TLS between the Cloudera Manager Server and Agents – three levels available**
 - Level 1 (good) – Encrypts communication between browser and CM and between agents and CM Server
 - Level 2 (better) – Adds strong verification of CM Server certificate
 - Level 3 (Best) – Authentication of agents to the CM Server using certs
- **Configure SSL encryption for CDH services (HDFS, YARN, etc.)**
 - Example: Encrypt data transferred between DataNodes and between DataNodes and clients
 - Recommendation: enable Kerberos on the cluster first
 - Steps to configure encryption differs for each CDH service
 - See Cloudera documentation for details

Apache Sentry

- **Sentry is a plugin for enforcing role-based authorization for Hive and Impala**
 - Stores authorization policy metadata
 - Provides clients with concurrent and secure access to this metadata
 - Database-style authorization for Hive, Impala, and Search
- **Sentry requirements**
 - CDH 4.3 or later
 - HiveServer2 and Hive Metastore running with strong authentication
 - Impala 1.2.1 or later running with strong authentication
 - Kerberos enabled on the cluster
- **Use Cloudera Manager's 'Add a Service' wizard to install Sentry**
 - Requires HDFS and ZooKeeper roles
 - Deploys a 'Sentry Server' daemon



Sentry Access Control Model

- **What does Sentry control access to?**
 - Server, Database, Table, View
- **Who can access Sentry-controlled objects?**
 - Users in a Sentry role
 - Sentry roles = one or more groups
- **How can role members access Sentry-controlled objects?**
 - Read operations (SELECT privilege)
 - Write operations (INSERT privilege)
 - Metadata definition operations (ALTER privilege)
- **Option to synchronize HDFS ACLs and Sentry permissions**
 - Set user access permissions once and securely share the same HDFS data with other components (Pig, Spark, MR, and HDFS clients, etc.)
 - Example use case: importing data into Hive using Sqoop

Chapter Topics

Hadoop Security

- Why Hadoop Security Is Important
- Hadoop's Security System Concepts
- What Kerberos Is and How it Works
- Securing a Hadoop Cluster with Kerberos
- Other Security Topics
- **Conclusion**

Essential Points

- Kerberos is the primary technology for enabling authentication security on the cluster
 - Manual configuration requires many steps
 - We recommend using Cloudera Manager to enable Kerberos
- Encryption can be enabled at the filesystem level, HDFS level, and the network level
- Sentry enables security for Hive and Impala



Managing Resources

Chapter 13



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- **Managing Resources**
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- Conclusion

Managing Resources

In this chapter, you will learn

- **How to allocate resources between various Hadoop frameworks using Static Service Pools**
- **How to configure resource availability and scheduling for YARN applications and Impala queries using Dynamic Resource Pools and Admission Control**
- **Recommended YARN memory and CPU settings**

Resource Management Overview

- **Hadoop applications compete for cluster resources**
- **Common objectives of resource management**
 - Guarantee the completion of critical workloads in reasonable timeframe
 - Coordinate cluster resource usage between competing groups of users
 - Prevent users from depriving other users cluster access
- **Tools available for managing Hadoop resources**
 - Linux Control Groups (cgroups)
 - Used to allocate resources between frameworks
 - Configurable using Static Service Pools
 - Schedulers and Admission Controls
 - For prioritizing when and where YARN applications and Impala queries run
 - Configurable using Dynamic Resource Pools and other settings

Chapter Topics

Managing Resources

- **Configuring cgroups with Static Service Pools**
- The Fair Scheduler
- Configuring Dynamic Resource Pools
- YARN Memory and CPU Settings
- Impala Query Scheduling
- Conclusion
- Hands-On Exercise: Using The Fair Scheduler

Linux Control Groups (cgroups)

- **Linux control groups (cgroups) are a kernel feature for setting restrictions on Linux processes**
 - Useful for isolating computation frameworks from one another
 - Set memory limits, CPU shares, and I/O weight
- **RHEL 6 and CentOS 6 support cgroups (RHEL 5 and CentOS 5 do not)**
 - See the Cloudera documentation for cgroup support on other distros
- **Enable cgroup-based resource management of Hadoop processes in Cloudera Manager**

The screenshot shows the Cloudera Manager interface. The top navigation bar includes Home, Clusters, Hosts, Diagnostics, Audits, Charts, and Backup. Below this, the Hosts tab is selected. The main content area has tabs for Hosts, Status, Configuration, Templates, Disks Overview, and Parcels. Under Configuration, there is a search bar and a note: "Settings marked will not take effect until after the next cluster restart". A table lists configuration properties:

Category	Property	Value
Parcels	Enable Cgroup-based Resource Management	<input type="checkbox"/> default value
Other		
Advanced		
Resource Management		

When to Use cgroups

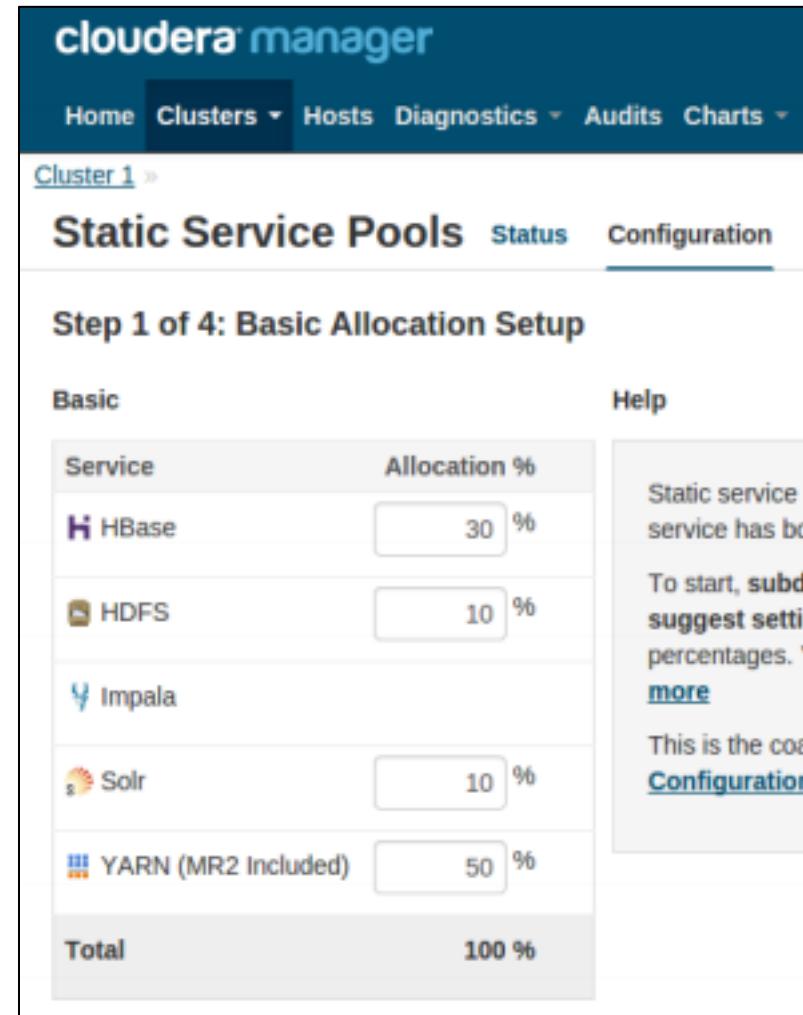
- **Cloudera recommends configuring cgroups when YARN and non-YARN services will share cluster node resources**
- **Example scenario:**
 - MapReduce and/or Spark running on YARN
 - Impala (not running on YARN) and perhaps HBase, HDFS, Flume, or Search services also on the cluster
- **Configure cgroups to ensure that one service cannot overuse cluster resources**
 - Isolate services from one another on the cluster

Configuring cgroup Parameters

- **The following resource limits can be configured after enabling cgroups**
 - **Memory Hard Limit**
 - If a process exceeds this limit, the kernel swaps out some of the process's memory; if it cannot do so, the process will be killed
 - **Memory Soft Limit**
 - When memory contention exists on the host, the OS targets the process to not exceed this limit
 - **CPU Shares**
 - When CPU contention exists on the host, processes with higher CPU shares will be given more CPU time
 - **I/O Weight**
 - Specify the proportion of I/O access available to the read requests performed by a process
- **Rather than manually configure these settings, configure a Static Service Pool for the cluster**

Static Service Pools (1)

- The Static Service Pools wizard configures cgroups for you
- Static Service Pools partition cluster resources across Hadoop services
 - Isolate services in the cluster from one another
 - Limits impact of a given CDH service on other services
- Example use case:
 - Allocate host resources across HDFS, YARN, Impala, Solr, and HBase services on the same cluster



Static Service Pools (2)

- Services are allocated a static percentage of total resources
 - CPU, memory, and I/O weight
- CM computes recommended configurations for each of a service's worker roles
 - Corresponding to the percentage assigned to each service
- The wizard provides an option to change cgroup settings directly

The screenshot shows the Cloudera Manager interface for managing Static Service Pools. The top navigation bar includes Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. The main content area is titled "Static Service Pools" and "Step 2 of 4: Change Settings Directly". On the left, there's an "Advanced" section showing allocation percentages for various services: HDFS (10%), Impala (20%), Solr (10%), and YARN (60%). The total allocation is 100%. To the right, under "Details", there are two checkboxes: "Use CGroups for Resource Management" (checked) and "Always Use Linux Container Executor" (checked). Below these are tables for "Cgroup CPU Shares" and "YARN (MR2 Included): NodeManager". The "Cgroup CPU Shares" table shows 3 hosts with a value of 1320, a subtotal of 3960, and a total of 3960 at 100.0% allocation. The "YARN (MR2 Included): NodeManager" table shows the same information. At the bottom, there are "Back" and "Continue" buttons, along with a page navigation bar showing pages 1, 2, 3, and 4.

Chapter Topics

Managing Resources

- Configuring cgroups with Static Service Pools
- **The Fair Scheduler**
- Configuring Dynamic Resource Pools
- YARN Memory and CPU Settings
- Impala Query Scheduling
- Conclusion
- Hands-On Exercise: Using The Fair Scheduler

The YARN Scheduler

- **The ResourceManager's scheduling component (the YARN Scheduler) is responsible for assigning available resources to YARN applications**
 - The scheduler decides *where* and *when* containers will be allocated to applications
 - Based on requirements of each application
 - Containers granted specific resources
 - memory, CPU
- **Administrators define a scheduling policy that best fits requirements**
 - For example, a policy that allocates resources equally among YARN applications
- **The scheduling policy establishes rules for resource sharing**
 - Rules are well-defined
 - Rules form the basis for application start and completion expectations

Scheduler Types

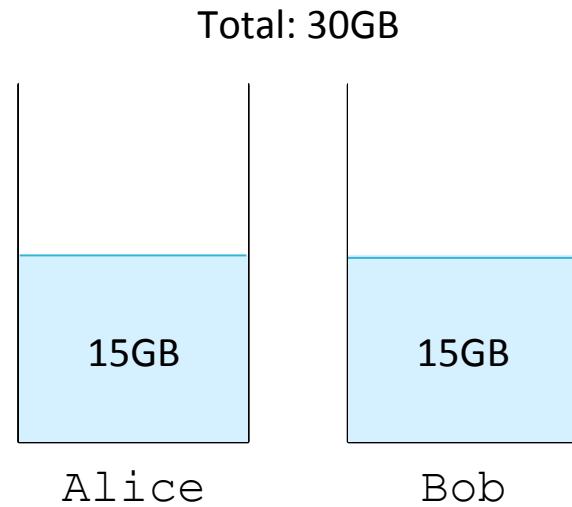
- **YARN supports the following schedulers:**
 - FIFO Scheduler
 - Resources allocated based on time of arrival
 - Capacity Scheduler
 - Resources allocated to pools
 - FIFO scheduling within each pool
 - Fair Scheduler
 - Resources allocated to weighted pools
 - Fair sharing within each pool
- **The Fair Scheduler is the default YARN scheduler in CDH 5**
 - The `yarn.resourcemanager.scheduler.class` property specifies the scheduler in use

The Fair Scheduler

- **The Fair Scheduler is the YARN Scheduler that Cloudera recommends for production clusters**
- **The Fair Scheduler organizes YARN applications into pools**
 - Pools are also known as a *queues* in YARN terminology
 - Each user get a pool named after the user (by default)
 - Resources are divided fairly between the pools (by default)
- **Fair Scheduler characteristics**
 - Allows resources to be controlled proportionally
 - Promotes efficient utilization of cluster resources
 - Promotes fairness between schedulable entities
 - Allows short interactive and long production applications to co-exist
 - Awards resources to pools that are most underserved
 - Gives a container to the pool that has the fewest resources allocated

Fair Scheduler Pools

- Each application is assigned to a *pool*
- All pools descend from the root pool
- Pools can be nested as subpools in which case siblings share the parent's resources
- Physical resources are not bound to any specific pool
- Pools can be predefined or defined dynamically by specifying a pool name when you submit an application

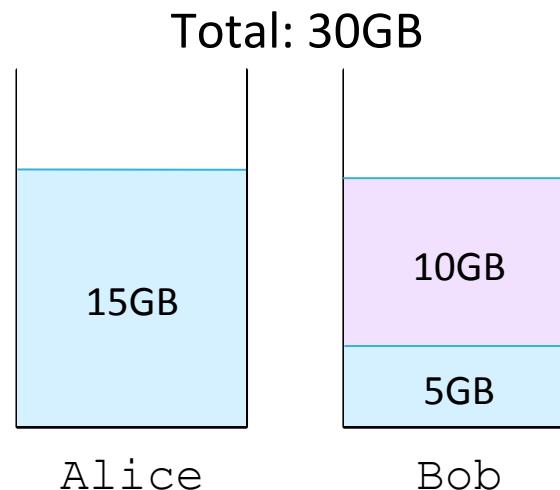


Determining the Fair Share

- **The fair share of resources assigned to the pool is based on**
 - The total resources available across the cluster
 - The number of pools competing for cluster resources
- **Excess cluster capacity is spread across all pools**
 - The aim is to maintain the most even allocation possible so every pool receives its fair share of resources
- **The fair share will never be higher than the actual demand**
- **Pools can use more than their fair share when other pools are not in need of resources**
 - This happens when there are no tasks eligible to run in other pools

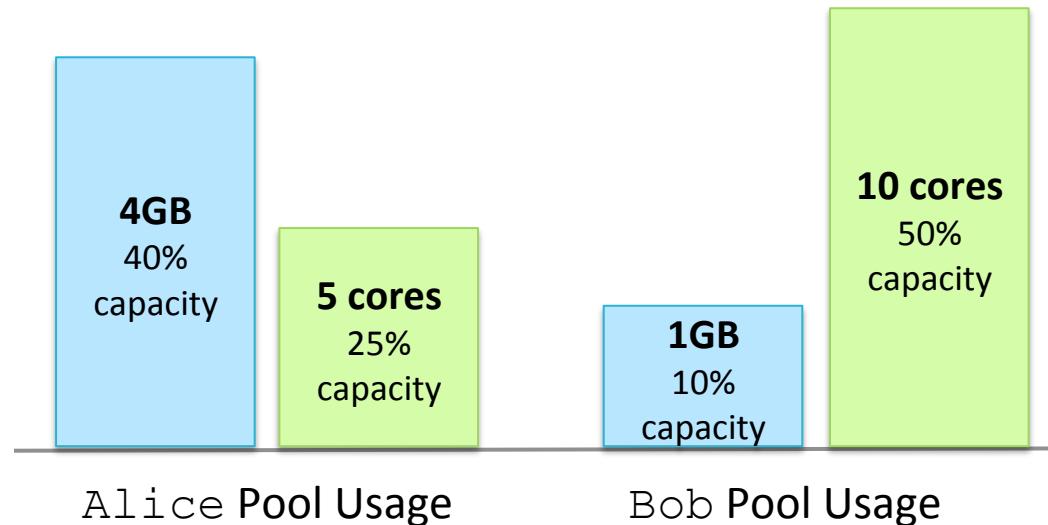
Single Resource Fairness

- **Fair scheduling with Single Resource Fairness**
 - Schedules applications on the basis of a single resource: memory
- **Example**
 - Two pools: Alice has been allocated 15GB, and Bob has 5GB
 - Both pools request a 10GB container of memory
 - Bob has less resources at the moment and will be granted the next 10GB that becomes available



Dominant Resource Fairness

- **Fair scheduling with Dominant Resource Fairness (*recommended*)**
 - Schedules applications on the basis of both memory and CPU
- **Example: A cluster has 10GB of total memory and 20 cores**
 - Pool Alice has containers granted for 4GB of memory and 5 cores
 - Pool Bob has containers granted for 1GB of memory and 10 cores
 - Alice will receive the next container because its 40% dominant share of memory is less than the Bob pool's 50% dominant share of CPU



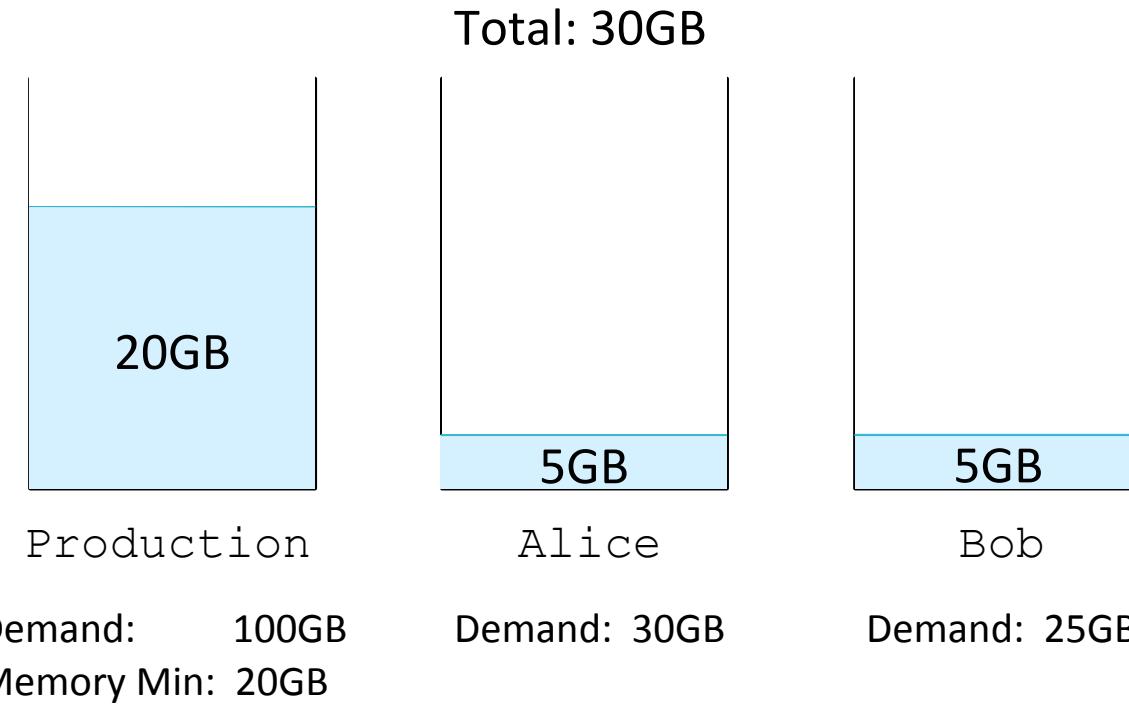
Minimum Resources

- A pool with minimum resources defined receives priority during resource allocation
- The minimum resources are the minimum amount of resources that must be allocated to the pool *prior* to fair share allocation
 - Minimum resources are allocated to each pool, assuming there is cluster capacity

Minimum Allocation Example

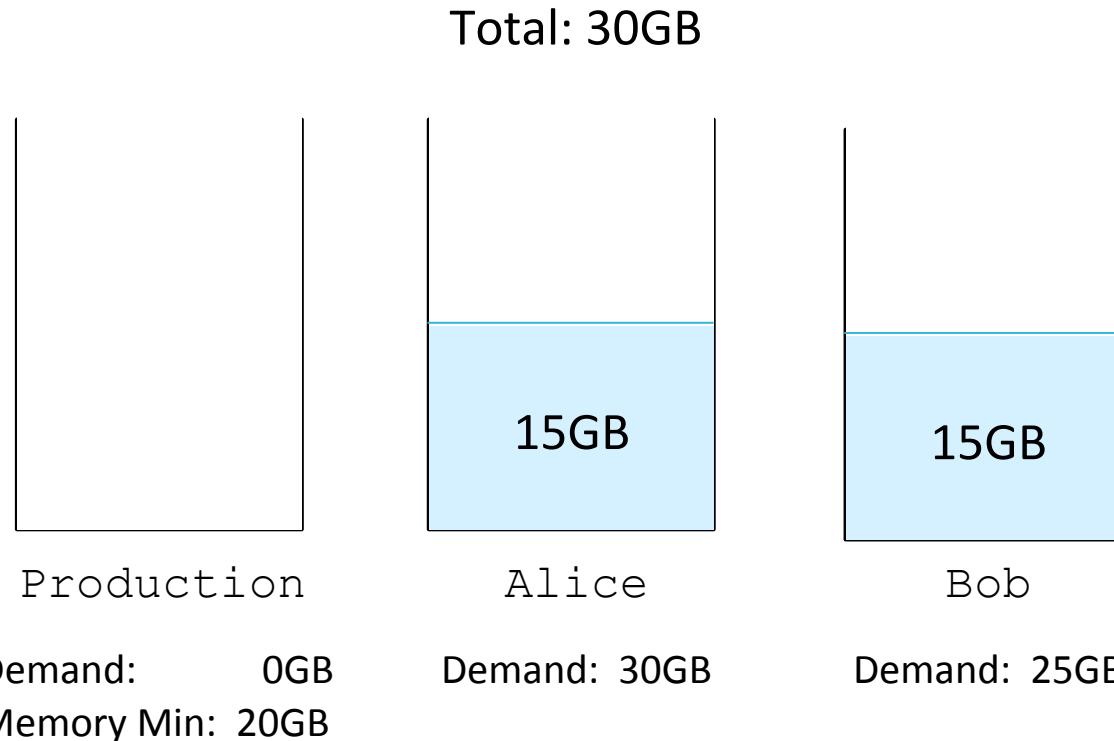
- **30GB memory available on the cluster**

- First, fill up the Production pool to the 20GB minimum guarantee
- Then distribute the remaining 10GB evenly across Alice and Bob



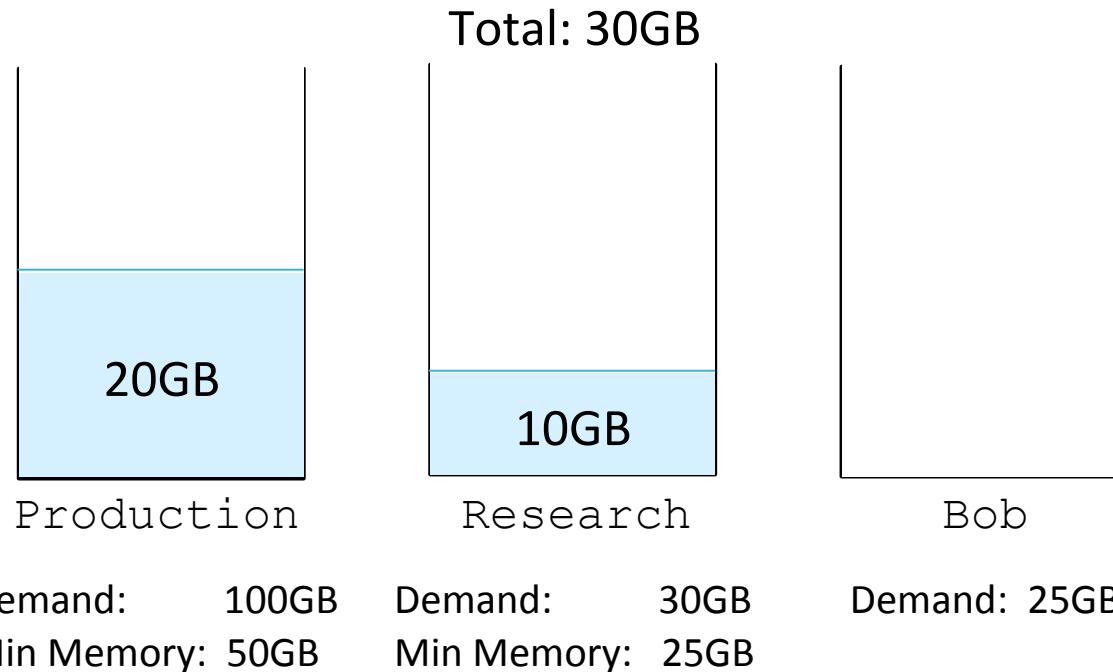
Minimum Allocation Example 2: Production Pool Empty

- **Production has no demand, so no resources are allocated to it**
- **All resources are allocated evenly between Alice and Bob**



Minimum Allocation Example 3: Min Memory Exceeds Resources

- Combined minimum memory requirements of Production and Research exceed capacity
- Resources are assigned proportionally based on defined minimum resources until available memory is exhausted
- No memory remains for pools without min memory defined (i.e., Bob)

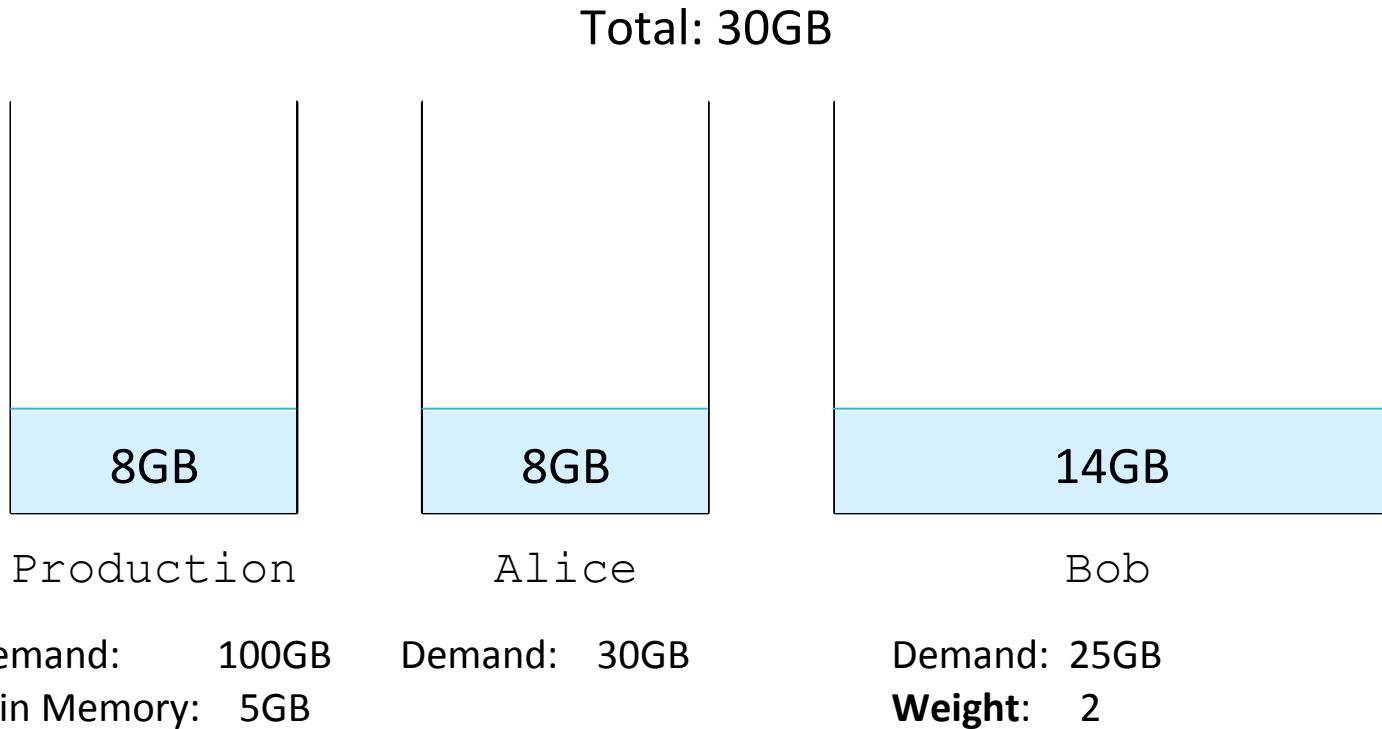


Pools with Weights

- Instead of (or in addition to) setting minimum resource requirements, pools can be assigned a *weight*
- Pools with higher weight receive more resources during allocation
- ‘Even water glass height’ analogy:
 - Think of the weight as controlling the ‘width’ of the glass

Pools with Weights Example: Pool With Double Weight

- Production is filled to minimum memory (5GB)
- Remaining 25GB is distributed across pools
- Bob pool receives twice the amount of memory during fair share allocation



Fair Scheduler Preemption (1)

- Recall that pools can use more than their fair share when other pools are not in need of resources
 - This resource sharing is typically a good thing
- There may be occasions when a pool needs its fair share back
 - Example: production applications must run within a set time period
 - Some pools must operate on an SLA (Service Level Agreement)
 - Solution: the preemption feature allows the cluster to be used by less critical applications while also ensuring important applications are not starved for too long
- If `yarn.scheduler.fair.preemption` is enabled:
 - the Fair Scheduler kills containers that belong to pools operating over their fair share beyond a configurable timeout
 - Pools operating below fair share receive those reaped resources
- CM default: preemption is not enabled

Fair Scheduler Preemption (2)

- **There are two types of preemption available**
 - Minimum share preemption
 - Fair share preemption
- **Preemption avoids killing a container in a pool if it would cause that pool to begin preempting containers in other pools**
 - This prevents a potentially endless cycle of pools killing one another's containers
- **Use fair share preemption conservatively**
 - Set the **Min Share Preemption Timeout** to the number of seconds a pool is under fair share before preemption should begin
 - Default is infinite

Options for Configuring the YARN Scheduler

- Two options are available for specifying how YARN applications share resources
 - 1. Manually configuring the YARN Scheduler
 - 2. Configure Dynamic Resource Pools (*recommended*)
 - Configures the YARN Scheduler for you

Chapter Topics

Managing Resources

- Configuring cgroups with Static Service Pools
- The Fair Scheduler
- **Configuring Dynamic Resource Pools**
- YARN Memory and CPU Settings
- Impala Query Scheduling
- Conclusion
- Hands-On Exercise: Using The Fair Scheduler

Dynamic Resource Pools Overview

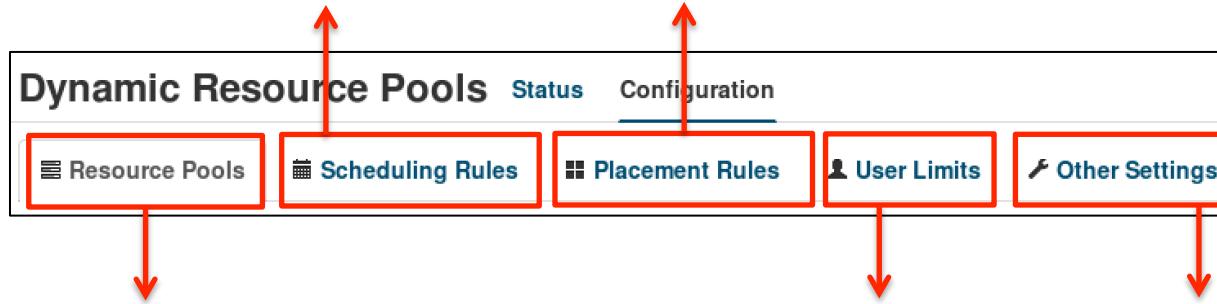
- A Dynamic Resource Pool (DRP) specifies a set of resources and a scheduling policy for ‘applications’ running in the pool
 - Applications can be YARN applications or Impala queries
- Useful for dynamically apportioning the percentage of cluster resources allocated to YARN and Impala
 - If Static Service Pool (cgroup) settings are in force, the dynamic apportionment will be *within* the percentage of resources granted to YARN or Impala
- Dynamic Resource Pools are the recommended method to configure the Fair Scheduler
- DRPs can also be used to configure Impala admission control or resource management for Impala
 - Discussed later in the chapter

Why Configure Dynamic Resource Pools?

- **Reasons to create Dynamic Resource Pools**
 - For calendar-based scheduling purposes
 - Create ‘Configuration Sets’ that can be configured to use different per-pool CPU and memory settings
 - To configure rules that determine in which pool an application will run
 - To set limits on how many applications a given user can run at one time
- **Dynamic Resource Pools provide a simple method for configuring the Fair Scheduler**
 - Allocate resources across pools
 - Specify min and max limits and weights

Dynamic Resource Pool Configuration Settings

- Set rules that determine in which pool an application will run
 - Some options include running apps in a pool named after the submitting user and allowing pool creation at runtime
- Configure an ordered list of calendar-based scheduling rules



- Resource Pools page
 - Configure default pool settings
 - Add a new resource pool
 - Displays settings for each existing pool Including weight, min/max vCores and memory, max running apps, and scheduling policy
- Set max number of applications a user can simultaneously run
- Configure other Fair Scheduler settings
 - For example: preemption

Dynamic Resource Pools – Scheduling Rules

- A “Configuration Set” defines the allocation of resources across pools that may be active at a given time
- Define Configuration Sets in the ‘Scheduling Rules’ tab
 - Example sets: Weekdays, Weekends
- Then when defining a pool, specify different settings for different times

Scheduling Rules

Configuration Set	
Weekdays	<input checked="" type="checkbox"/> Edit
Weekends	<input checked="" type="checkbox"/> Edit
default	

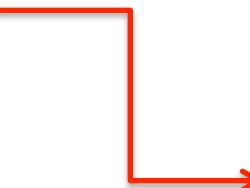
Resource Pools

Edit Resource Pool: Engineering

General YARN Submission Access Control Administration Access Control

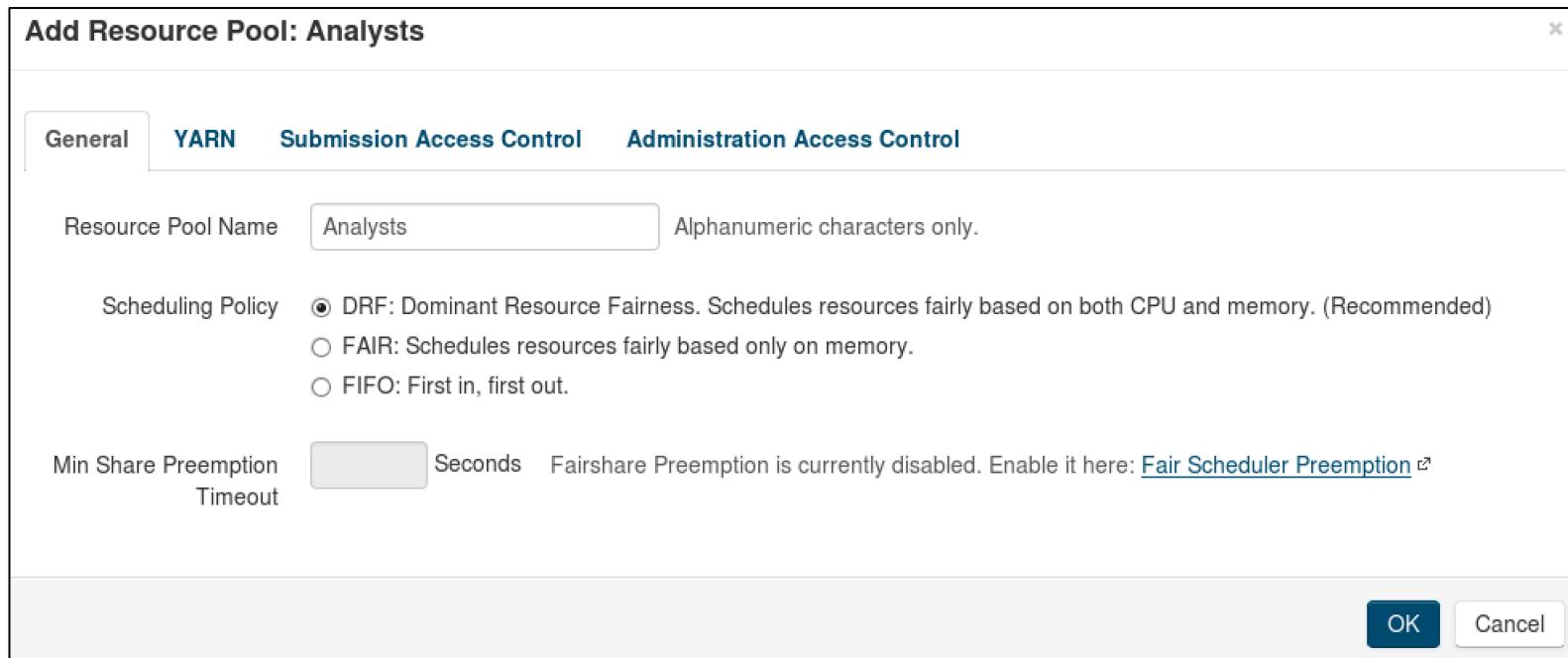
Multiple configuration sets allow you to specify different settings based on your schedule.

default	Weight	2 Share of resources
Weekdays	Virtual Cores (Min / Max)	6 / 24 The minimum and the maximum override weight settings. (optional)
Weekends	Memory (Min / Max)	16000 / 96000 MB



Adding a Dynamic Resource Pool (1)

- From the Clusters > ClusterName > Dynamic Resource Pools ‘Configuration’ tab, click ‘Add Resource Pool’
 - Give the pool a name
 - Set the Scheduling Policy for applications in the pool
 - If Fairshare Preemption is enabled, optionally set a timeout for it



Adding a Dynamic Resource Pool (2)

- In the YARN and/or Impala tab(s), options to set the weight, vCores and memory min/max, and max running apps settings for a configuration set

Edit Resource Pool: Production

General YARN **Impala** Submission Access Control Administration Access Control

Multiple configuration sets allow you to specify different settings based on your schedule.

default	Weight	1 Share of resources relative to other pools.
Weekdays	Virtual Cores (Min / Max)	2 / 8
Weekends		The minimum and the maximum number of virtual cores available to the pool. These override weight settings. (optional)
	Memory (Min / Max)	4000 / 64000 MB
		The minimum and the maximum amount of aggregate memory available to the pool. These override weight settings. (optional)
	Max Running Apps	8 A limit on the number of applications simultaneously running in a pool.

OK Cancel

- The ‘...Access Control’ tabs allow you to specify which users are allowed to submit apps to the pool and administer the pool

Dynamic Resource Pools – Placement Rules

- In the Dynamic Resource Pools ‘Placement Rules’ tab, configure rules determining in which pool an application will run
- **yarn.scheduler.fair.user-as-default-queue**
 - When true (default), the default pool for an application is *username*
 - If false, applications run in the default pool
- **yarn.scheduler.fair.allow-undeclared-pools**
 - If true (default), applications can declare a new pool at submission
 - If false, applications specifying unknown pools run in default pool
- Specify a pool in MapReduce using **–D mapreduce.job.queuename**
 - In Spark, use `spark.yarn.queue`
 - In Impala, use `SET REQUEST_POOL=<poolname>`

When Will an Application Run Within a Pool?

- The Fair Scheduler grants resources to a pool, but which application will get the next resources requested?
 - It depends on which of the three Fair Scheduler options was chosen for prioritizing applications *within* the pool:
 - Single Resource Fairness
 - Dominant Resource Fairness
 - FIFO
 - It can also depend on whether the Fair Scheduler has been configured to delay assignment of resources when a preferred rack or node is not available
 - This behavior is configured using the `yarn.scheduler.fair.locality.threshold.node` and `yarn.scheduler.fair.locality.threshold.rack` properties

Dynamic Resource Pools – User Limits

- When defining a pool you can set ‘Max Running Apps’ for the pool
- There is also the option to limit the number of applications specific users can run at the same time across pools
 - Define a limit for all users in “Default Settings”
 - Define a specific limit for an individual user

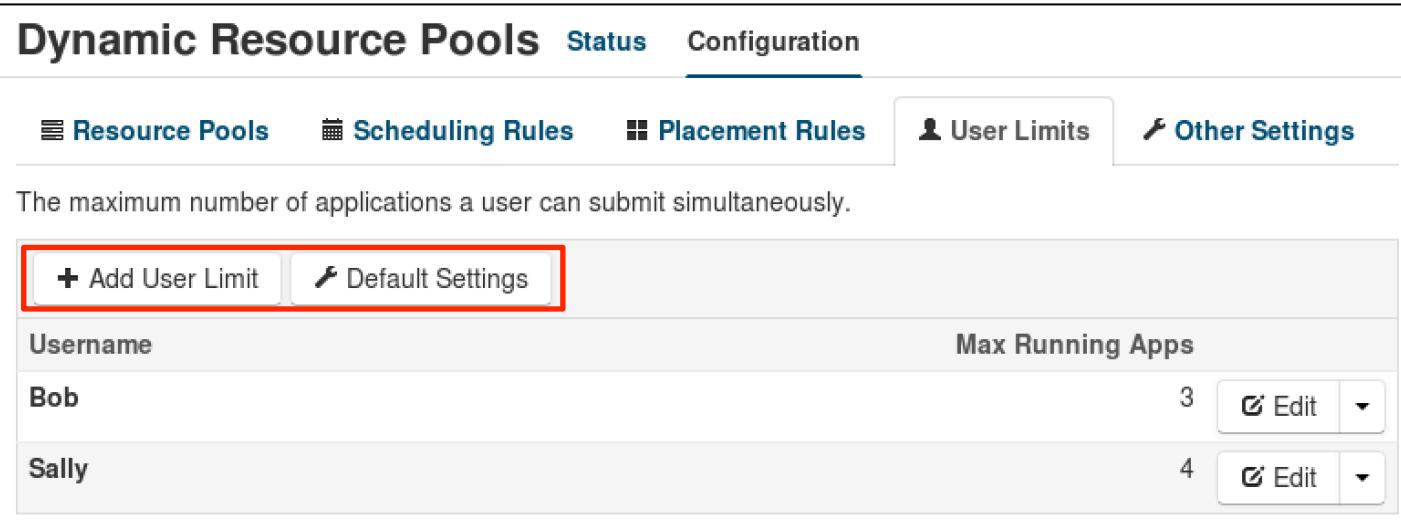
Dynamic Resource Pools Status Configuration

Resource Pools Scheduling Rules Placement Rules User Limits Other Settings

The maximum number of applications a user can submit simultaneously.

+ Add User Limit Default Settings

Username	Max Running Apps
Bob	3
Sally	4

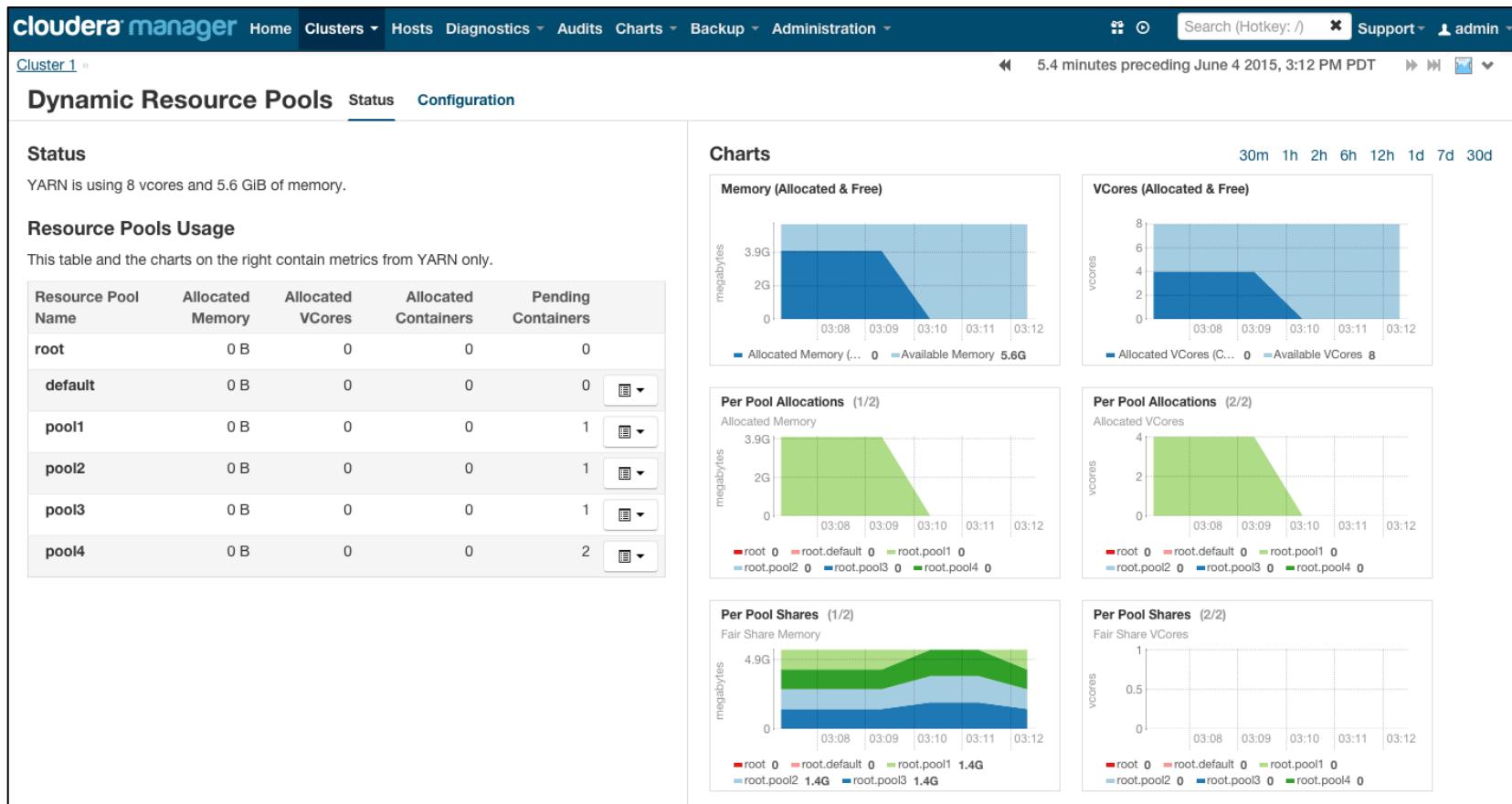


Fair Scheduler Settings Deployment

- The Fair Scheduler settings configured are viewable in JSON format in Cloudera Manager in the “Fair Scheduler Allocations” property
 - Found in YARN / Service-Wide / Advanced
- These settings are deployed by Cloudera Manager to a `fair-scheduler.xml` file on the ResourceManager host
 - The Fair Scheduler rereads this file every 10 seconds
- ResourceManager restart is *not* required when the file changes

Cloudera Manager - Fair Scheduler Related Charts

Cloudera Manager's Dynamic Resource Pools 'Status' page



Chapter Topics

Managing Resources

- Configuring cgroups with Static Service Pools
- The Fair Scheduler
- Configuring Dynamic Resource Pools
- **YARN Memory and CPU Settings**
- Impala Query Scheduling
- Conclusion
- Hands-On Exercise: Using The Fair Scheduler

Getting an Overview of Resource Utilization by Host

- The Host “Resources” tab view shows CPU and Memory usage
- Add up memory consumption of daemons
 - Does it approach or exceed memory available on the box?
 - Could explain swapping or OOM
- Add up vcore consumption of daemons
 - Does it approach or exceed available CPU on the box?
 - Could explain slow jobs
- The next few slides show how to adjust memory and CPU settings

The screenshot shows the Cloudera Manager interface for a host named 'horse'. The 'Resources' tab is selected. The CPU section displays the following data:

Service	Instance	Description	Approximate CPU
Flume	Agent	Agent CPU	1.0
HDFS	Balancer	HDFS Balancer, when running	1.0
HDFS	DataNode	DataNode CPU	0.5
HDFS	JournalNode	JournalNode CPU	1.0
Impala	Impala Catalog Server	Impala Catalog Server CPU	1.0
Impala	Impala Daemon	Impala Daemon CPU	1.0
Impala	Impala StateStore	Impala StateStore CPU	1.0
YARN (MR2 Included)	NodeManager	NodeManager process	0.5
YARN (MR2 Included)	NodeManager	NodeManager MR Containers	2.0
YARN (MR2 Included)	ResourceManager	ResourceManager CPU	1.0
ZooKeeper	Server	Server CPU	1.0

The Memory section displays the following data:

Service	Instance	Description	Bytes
Flume	Agent	Java Heap Size of Agent in Bytes (+ 30% Overhead)	65.0 MB
HDFS	Balancer	Java Heap Size of Balancer in Bytes (+ 30% Overhead)	275.6 MB
HDFS	DataNode	Java Heap Size of DataNode in Bytes (+ 30% Overhead)	1.3 GiB
HDFS	DataNode	Maximum Memory Used for Caching	4.0 GiB
HDFS	JournalNode	Java Heap Size of JournalNode in Bytes (+ 30% Overhead)	332.8 MB
Impala	Impala Catalog Server	Fixed Memory Size of Catalog Server in Bytes	150.0 MB
Impala	Impala Daemon	Impala Daemon Memory Limit	256.0 MB
Impala	Impala StateStore	Fixed Memory Size of StateStore in Bytes	64.0 MB
YARN (MR2 Included)	NodeManager	Java Heap Size of NodeManager in Bytes (+ 30% Overhead)	275.6 MB
YARN (MR2 Included)	NodeManager	Container Memory	1.0 GiB
YARN (MR2 Included)	ResourceManager	Java Heap Size of ResourceManager in Bytes (+ 30% Overhead)	275.6 MB
ZooKeeper	Server	Java Heap Size of ZooKeeper Server in Bytes (+ 30% Overhead)	65.0 MB

YARN – Resource Allocation: Worker Node Configuration

yarn.nodemanager.resource.memory-mb

Set in YARN / NodeManager Group / Resource Management

- Amount of RAM available on this host for YARN-managed tasks
- Recommendation: the amount of RAM on the host minus the amount needed for non-YARN-managed work (including memory needed by the DataNode daemon)
- Used by the NodeManagers

yarn.nodemanager.resource.cpu-vcores

Set in YARN / NodeManager Group / Resource Management

- Number of cores available on this host for YARN-managed tasks
- Recommendation: the number of physical cores on the host minus 1
- Used by the NodeManagers

YARN Scheduler Parameters

yarn.scheduler.minimum-allocation-mb
yarn.scheduler.minimum-allocation-vcores

Set in YARN / ResourceManager Group / Resource Management

- Minimum amount of memory and cpu cores to allocate for a container
- Task requests lower than these minimums will be set to these values
- CM Defaults: 1 GB, 1 vcore
- Memory recommendation: increase up to 4 GB depending on your developers' requirements
- Cores recommendation: keep the 1 vcore default
- Used by the ResourceManager

yarn.scheduler.increment-allocation-mb
yarn.scheduler.increment-allocation-vcores

Set in YARN / ResourceManager Group / Resource Management

- Tasks with requests that are not multiples of these increment-allocation values will be rounded up to the nearest increments
- CM Defaults: 512MB and 1 vcore

YARN – Resource Allocation: Memory Request for Containers

mapreduce.map.memory.mb and mapreduce.reduce.memory.mb

Set in YARN / Gateway Group / Resource Management

- Amount of memory to allocate for Map or Reduce tasks
- CM Default: 1 GB
- Recommendation: increase mapreduce.map.memory.mb up to 2 GB, depending on your developers' requirements. Also, set mapreduce.reduce.memory.mb to twice the mapper value.
- Used by clients and NodeManagers

yarn.app.mapreduce.am.resource.mb

Set in YARN / Gateway Group / Resource Management

- Amount of memory to allocate for the ApplicationMaster
- CM Default: 1 GB
- Recommendation: 1 GB, however you can increase it if jobs contain many concurrent tasks.
- Used by clients and NodeManagers

YARN – MapReduce Container Heap Size

yarn.app.mapreduce.am.command-opts

Set in YARN / Gateway Group

- Java options passed to the ApplicationMaster
- By Default Application Master gets 1GB of heap space
- Used when MapReduce ApplicationMasters are launched

mapreduce.map.java.opts

mapreduce.reduce.java.opts

Set in YARN / Gateway Group

- Java options passed to Mappers and Reducers
- Default is $-Xmx=200m$ (200MB of heap space)
- Recommendation: increase to a value from 1GB to 4GB, depending on the requirements from your developers
- Used when Mappers and Reducers are launched

YARN – Configure Resource Allocation and Process Size Properties

- The resource allocation properties do *not* determine the heap size for the ApplicationMaster, Mappers, and Reducers
- Be sure to adjust both the Java heap size properties *and* the resource allocation properties
- For example, if you specified
 - `yarn.nodemanager.resource.memory-mb = 8192`
 - `mapreduce.map.memory.mb = 4096`
 - And allowed `mapreduce.map.java.opts` to default
- Then the maximum heap size for Mappers would be 200MB
 - Because `mapreduce.map.java.opts` defaults to `-Xmx=200m`
- Recommendation: set the Java heap size for Mappers and Reducers to 75% of `mapreduce.map.memory.mb` and `mapreduce.reduce.memory.mb`

Summary of YARN Memory and CPU Default Settings

Service	Setting	Default
YARN Gateway	ApplicationMaster Memory	1GB
YARN Gateway	ApplicationMaster Java Max Heap Size	825MB
YARN Gateway	Map Task Memory	1GB
YARN Gateway	Map Task Max Heap Size	825MB
YARN Gateway	Reduce Task Memory	1GB
YARN Gateway	Reduce Task Max Heap Size	825MB
JobHistory Server	Java Heap Size	1GB
NodeManager	Java Heap Size	1GB
NodeManager	Container Memory	8GB
NodeManager	Container Virtual CPU Cores	8
ResourceManager	Java Heap Size	1GB
ResourceManager	Container Memory Min	1GB
ResourceManager	Container Memory Max	64GB
ResourceManager	Container Virtual CPU Cores Min	1
ResourceManager	Container Virtual CPU Cores Max	32

YARN Tuning Recommendations

- **Inventory the vcores, memory, and disks available on each worker node**
- **Calculate the resources needed for other processes**
 - Reserve 3GB or 20% of total memory for the OS
 - Reserve resources for any non-Hadoop applications
 - Reserve resources for other any Hadoop components
 - HDFS caching (if configured), NodeManager, DataNode
 - Impalad, HBase RegionServer, Solr, etc.
- **Grant the resources not used by the above to your YARN containers**
- **Configure the YARN scheduler and application framework settings**
 - Based on the worker node profile determined above
 - Determine the number of containers needed to best support YARN applications based on the type of workload
 - Monitor usage and tune estimated values to find optimal settings

Chapter Topics

Managing Resources

- Configuring cgroups with Static Service Pools
- The Fair Scheduler
- Configuring Dynamic Resource Pools
- YARN Memory and CPU Settings
- **Impala Query Scheduling**
- Conclusion
- Hands-On Exercise: Using The Fair Scheduler

Impala Admission Control (1)

- **Impala Admission Control is a feature that provides a simple and robust way to manage per-pool resource utilization**
 - Enable this feature if the cluster is underutilized at some times and overutilized at others
- **Imposes resource scheduling on concurrent SQL queries without using the YARN scheduler**
 - Limits the max number of concurrent Impala queries and memory used by those queries in a given pool
 - Helpful for avoiding OOM issues on busy clusters
- **Also allows you to limit the max number of queued (waiting) queries**
 - Additional queries queued until earlier ones finish
 - As other queries finish queued queries allowed to proceed

Impala Admission Control (2)

- **Impala Admission Control is turned off by default**
- **Options for configuring Admission Control for Impala**
 - ‘Enable Dynamic Resource Pools’ property for Impala
 - The Recommended approach
 - If true, you can configure Impala Admission Control using DRPs
 - This is the recommended approach
 - ‘Enable Impala Admission Control’ property for Impala
 - If true, this will make Impala use its own single pool configuration

Impala Admission Control (3)

- You can specify different sets of pool names and allocation options for your Impala and YARN deployments
- Use case
 - You are setting up a multi-tenant cluster with cgroups
 - You want to allocate a certain portion of the cluster's resources to Impala and another portion to MapReduce

Impala on YARN

- In CDH 5, Impala can optionally be configured to use YARN for resource management
 - Impala estimates the resource needs for a query and makes a request to YARN to allocate resources
 - Requires the Impala Llama ApplicationMaster role instance
 - Llama acts as the intermediary for Impala to request resources from the YARN Resource Manager
- Impala on YARN is supported, but is not currently recommended for most use cases
 - This approach *may* result in higher latency than simply using cgroups and Admission Control to manage Impala

Summary of Dynamic Allocation Options Available in CDH 5

Type of Resource Management	YARN Independent	Impala Independent	YARN and Impala Independent	YARN and Impala Integrated
Configuration used	YARN scheduler	Impala Admission Control	YARN scheduler <i>and</i> Impala Admission Control	YARN scheduler (with Llama)
Feature In the Pool	Managed By			
vCores	YARN	cgroups	YARN	cgroups and YARN
Memory	YARN	Impala	YARN for YARN applications, Impala for pools running Impala queries	YARN (for both YARN apps and Impala queries)
Running applications	YARN	n/a	YARN	YARN
Scheduling policy	YARN	Impala	YARN	YARN
Limit the number of running and queued queries	n/a	Impala	Impala	Impala

When Impala and MapReduce Co-Exist on the Cluster

- **Recommendations for when Impala and MapReduce co-exist on the cluster:**
 - Set cgroup resource limits between MapReduce/YARN and Impala
 - Configure a ‘Static Service Pool’
 - Percentage allocation to each service will depend on how the cluster will be used
 - Configure Admission Control and query queuing
 - To manage concurrency and memory usage for Impala

Chapter Topics

Managing Resources

- Configuring cgroups with Static Service Pools
- The Fair Scheduler
- Configuring Dynamic Resource Pools
- YARN Memory and CPU Settings
- Impala Query Scheduling
- **Conclusion**
- Hands-On Exercise: Using The Fair Scheduler

Essential Points

- **cgroups allow you to partition cluster resources between different Hadoop frameworks such as MapReduce and Impala**
 - Configurable by defining Static Service Pools,
- **The Fair Scheduler is the default scheduler in CDH 5**
 - Allow resources to be controlled proportionally
 - Ensure that a cluster is used efficiently
 - Configurable by defining Dynamic Resource Pools
- **The Fair Scheduler is the most commonly-used scheduler**
 - Efficient when pool utilization varies
 - Delayed task assignment leads to better data locality

Chapter Topics

Managing Resources

- Configuring cgroups with Static Service Pools
- The Fair Scheduler
- Configuring Dynamic Resource Pools
- YARN Memory and CPU Settings
- Impala Query Scheduling
- Conclusion
- **Hands-On Exercise: Using The Fair Scheduler**

Hands-On Exercise: Using The Fair Scheduler

- In this Hands-On Exercise, you will run MapReduce jobs in different pools and observe how the Fair Scheduler handles the jobs
- Please refer to the Hands-On Exercise manual



Cluster Maintenance

Chapter 14



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- **Cluster Maintenance**
- Cluster Monitoring and Troubleshooting
- Conclusion

Cluster Maintenance

In this chapter, you will learn:

- **How to check the status of HDFS**
- **How to copy data between clusters**
- **How to add and remove nodes**
- **How to rebalance the cluster**
- **How to take HDFS snapshots**
- **How to upgrade your cluster**

Chapter Topics

Cluster Maintenance

- **Checking HDFS Status**
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

Checking for Corruption in HDFS (1)

- **hdfs fsck checks for missing or corrupt data blocks**
 - Unlike system fsck, does not attempt to repair errors
- **Can be configured to list all files**
 - Also all blocks for each file, all block locations, all racks
- **Examples:**

```
hdfs fsck /  
hdfs fsck / -files  
hdfs fsck / -files -blocks  
hdfs fsck / -files -blocks -locations  
hdfs fsck / -files -blocks -locations -racks
```

Checking for Corruption in HDFS (2)

- **Good idea to run `hdfs fsck` regularly**
 - Choose a low-usage time to run the check
- **`-move` option moves corrupted files to `/lost+found`**
 - A corrupted file is one where all replicas of a block are missing
- **`-delete` option deletes corrupted files**

Using dfsadmin (1)

- The **hdfs dfsadmin** command provides a number of administrative features including:
- List information about HDFS on a per-datanode basis

```
$ hdfs dfsadmin -report
```

- Re-read the **dfs.hosts** and **dfs.hosts.exclude** files

```
$ hdfs dfsadmin -refreshNodes
```

Using dfsadmin (2)

- Manually set the filesystem to ‘safe mode’
 - NameNode starts up in safe mode
 - Read-only – no changes can be made to the metadata
 - Does not replicate or delete blocks
 - Leaves safe mode when the (configured) minimum percentage of blocks satisfy the minimum replication condition

```
$ hdfs dfsadmin -safemode enter  
$ hdfs dfsadmin -safemode leave
```

- Can also block until safe mode is exited
 - Useful for shell scripts
 - hdfs dfsadmin -safemode wait

Using dfsadmin (3)

- Save the NameNode metadata to disk and resets the edit log
 - Must be in safe mode

```
$ hdfs dfsadmin -saveNamespace
```

- Allow an HDFS directory to be snapshotted

```
$ hdfs dfsadmin -allowSnapshot <directory_name>
```

Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- **Hands-On Exercise: Breaking The Cluster**
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

Hands-On Exercise: Breaking the Cluster

- In this Hands-On Exercise, you will introduce some problems into the cluster
- Please refer to the Hands-On Exercise Manual

Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- **Copying Data Between Clusters**
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

Copying Data

- Hadoop clusters can hold massive amounts of data
- A frequent requirement is to back up the cluster for disaster recovery
- Ultimately, this is not a Hadoop problem!
 - It's a 'managing huge amounts of data' problem
- Cluster could be backed up to tape etc. if necessary
 - Custom software may be needed

Copying Data with `distcp`

- **`distcp` copies data within a cluster, or between clusters**
 - Used to copy large amounts of data
 - Turns the copy process into a MapReduce job
- **Copies files or entire directories**
 - Files previously copied will be skipped
 - Note that the only check for duplicate files is that the file's name, size, and checksum are identical

distcp Examples

- **Copying data from one cluster to another**

- hadoop distcp hdfs://cluster1_nn:8020/path/to/src \
hdfs://cluster2_nn:8020/path/to/dest

- **Copying data within the same cluster**

- hadoop distcp /path/to/src /path/to/dest

- **Copying data from one cluster to another when the clusters are running different versions of Hadoop**

- HA HDFS example using HttpFS

- hadoop distcp hdfs://mycluster/path/to/src \
webhdfs://httpfs-svr:14000/path/to/dest

- Non-HA HDFS example using WebHDFS

- hadoop distcp hdfs://cluster1_nn:8020/path/to/src \
webhdfs://cluster2_nn:50070/path/to/dest

Copying Data: Best Practices

- **In practice, many organizations do not copy data between clusters**
- **Instead, they write their data to two clusters as it is being imported**
 - This is often more efficient
 - Not necessary to run all MapReduce jobs on the backup cluster
 - As long as the source data is available, all derived data can be regenerated later

Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- **Adding and Removing Cluster Nodes**
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

Administering Clusters and Hosts

- **Add a Cluster**

- Cloudera Manager allows you to manage multiple clusters

- **Add hosts to a cluster**

- Host(s) will be managed by Cloudera Manager
 - Use the ‘Add Hosts’ wizard
 - Installs the Cloudera Manager Agent
 - Optionally installs Oracle JDK
 - Installs CDH and any optional packages or parcels

- **After adding a host and installing CDH**

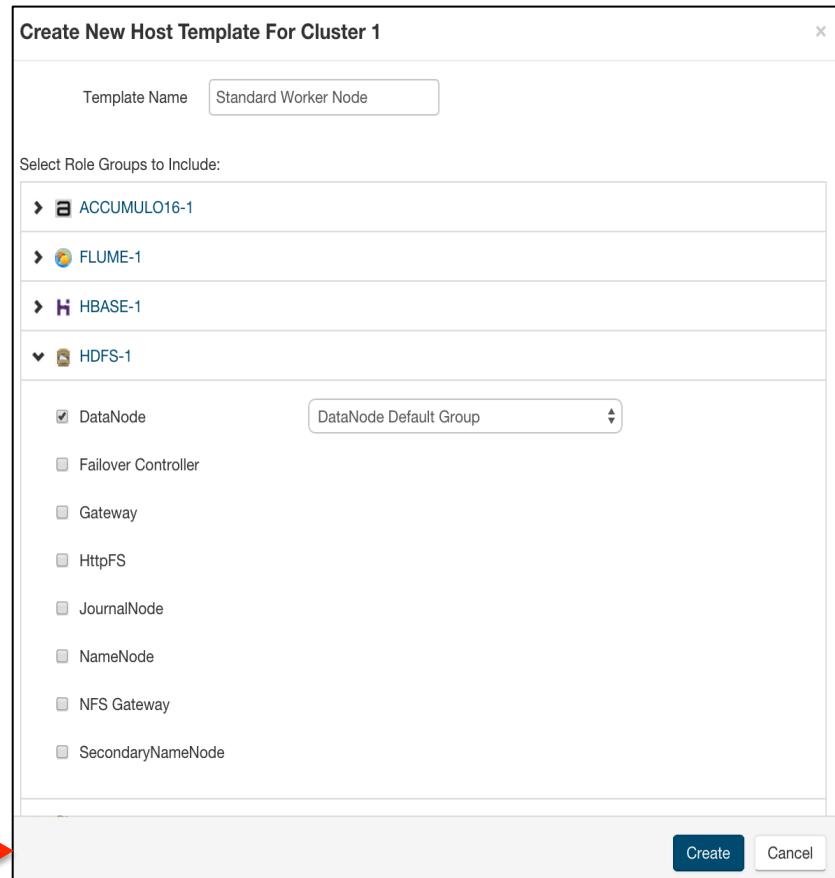
- Add roles or services to the host one at a time
 - or apply a ‘Host Template’

Host Templates

- Host Templates makes it easier to configure new hosts
- First create a ‘Host Template’ to define a set of ‘Role Groups’
- Then apply the Host Template to a host or set of hosts
 - Example: for hosts with the same hardware and services

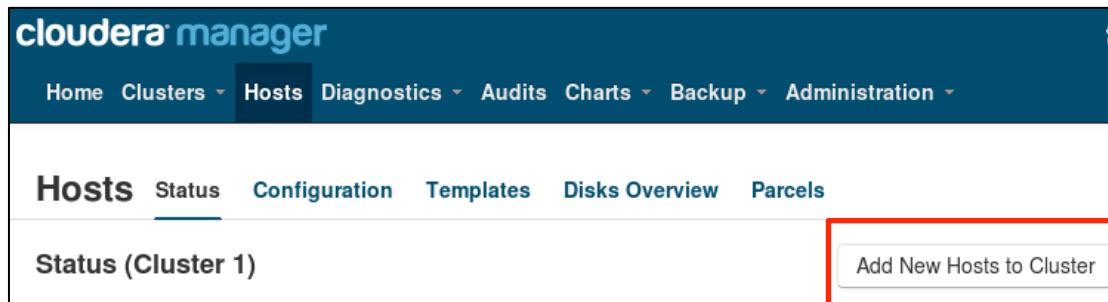
The screenshot shows the Cloudera Manager interface with the following details:

- Header: cloudera manager
- Navigation bar: Home, Clusters, Hosts (selected), Diagnostics, Audits, Charts, Backup, Administration.
- Main menu: Hosts, Status, Configuration, Templates (underlined), Disks Overview, Parcels.
- Section: Host Templates
- Sub-section: Host Templates for Cluster 1
- Action button: + Create... (highlighted with a red box and connected by a red arrow to the 'Create' button in the modal).



Adding Hosts to a Cluster

- Verify TLS encryption or authentication for Cloudera Manager Agents on the cluster is disabled (if not, temporarily disable first)
- Launch the ‘Add Hosts Wizard’



- Cloudera Manager will install CDH and the Cloudera Manager agent on the host(s) you identify
- After agent is installed add CDH roles or services, or apply a host template
- Re-enable TLS encryption if it was previously disabled
- Recommended: specify the rack(s) where the added host(s) are located

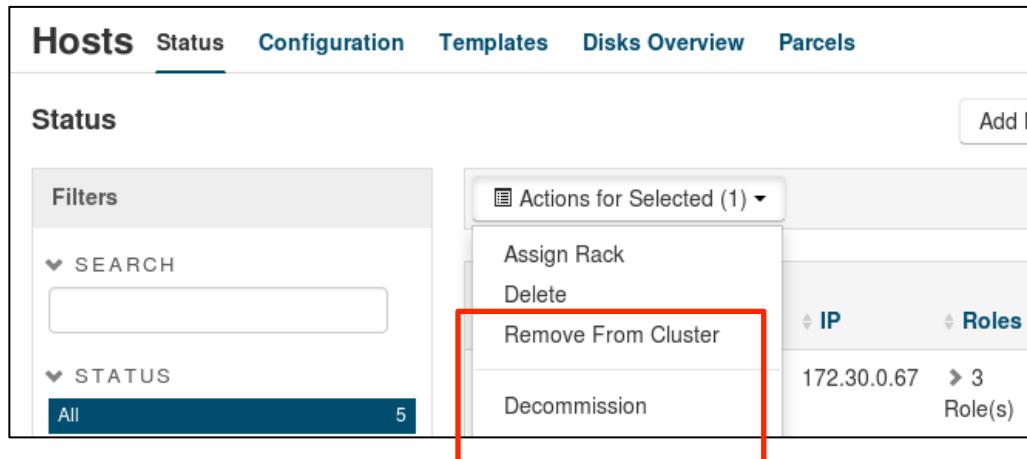
Adding Hosts: Points to Note

- **A NameNode will not ‘favor’ a new node added to the cluster**
 - It will not prefer to write blocks to the node rather than to other nodes
- **This is by design**
 - The assumption is that new data is more likely to be processed than older data
 - If all new blocks were written to the new node, this would impact data locality for applications
 - Would also create a ‘hot spot’ when writing new data to the cluster

Removing Hosts from a Cluster

■ Remove hosts from the cluster

- From Cloudera Manager’s Hosts page, select the node(s)
- Choose the appropriate option from the “Actions for Selected” menu
 - “Decommission” – host will no longer be managed by Cloudera Manager
 - “Remove from Cluster” – removes the host from the cluster, but keeps the host available to Cloudera Manager



Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- **Rebalancing the Cluster**
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

Cluster Rebalancing (1)

- HDFS DataNodes on cluster can become ‘unbalanced’
 - Some nodes have much more data on them than others
 - Example: add a new node to the cluster
 - Even after adding files to HDFS, new node will have far less data than others. During application processing, this node will use much more network bandwidth as it retrieves data from other nodes
- Clusters can be rebalanced using the HDFS ‘Rebalance’ option

The screenshot shows the Cloudera Manager interface for Cluster 1. The top navigation bar includes links for 'Status', 'Instances', 'Configuration', 'Commands', 'Audits', 'File Browser', and 'Logs'. The 'Actions' dropdown menu is open, listing options: Start, Stop, Restart, Rolling Restart, Rename, Enter Maintenance Mode, Rebalance (which is highlighted with a red box), and Deploy Client Configuration.

HDFS Summary

- Configured Capacity: 33.3 GB/86.5 GiB
- Quick Links: Replication, Reports, NameNode Web UI (Active)
- Event Search: Alerts, Critical, All

Status Summary

- SecondaryNameNode: 1 Good Health

Charts

HDFS Capacity

30m 1d

46.6G bytes

Thu 05 Sat 07

Configured... 86.5G ... 1.7G

Cluster Rebalancing (2)

- Balancer reviews data block placement on nodes and adjusts blocks to ensure all nodes are within the ‘Rebalancing Threshold’
 - Default: 10%
- ‘Used space to total capacity’ ratio *on each DataNode* will be brought to within the threshold of ‘used space to total capacity’ ratio *on the cluster*
- Balancer does not balance between individual volumes on a single DN
- Balancer bandwidth usage can be controlled
 - `dfs.datanode.balance.bandwidthPerSec`
 - Default: 10MB
 - Recommendation: approx. 0.1 x network speed
 - e.g., for a 1Gbps network, 10MB/sec
 - Note: balancer bandwidth can also be set for the current session
 - e.g. `dfsadmin - setBalancerBandwidth 10485760`

When To Rebalance

- **Rebalance immediately after adding new nodes to the cluster**
 - The Balancer must be manually invoked
 - Does not run automatically even if the Rebalance Threshold is exceeded
- **Rebalance during non-peak usage times**
 - Rebalancing does not interfere with any existing applications
 - However, it does use bandwidth

Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- **Hands-On Exercise: Verifying The Cluster's Self-Healing Features**
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

Hands-On Exercise: Verifying The Cluster's Self-Healing Features

- In this Hands-On Exercise, you will verify that the cluster has recovered from the problems you introduced in the last exercise
- Please refer to the Hands-On Exercise Manual

Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- **Directory Snapshots**
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

HDFS Directory Snapshots

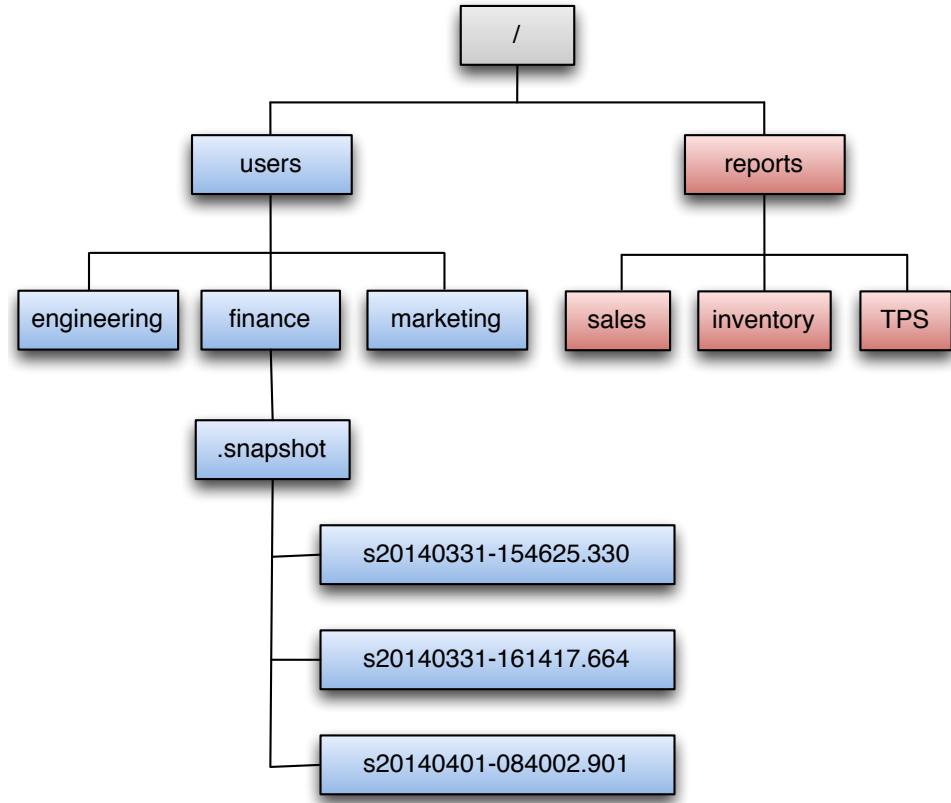
- A Snapshot is a read-only copy of an HDFS directory at a point in time

- Useful for data backup, disaster recovery
 - Also provides an option to snapshot the entire HDFS filesystem

- Snapshots appear on the filesystem as read-only directories

- Data is not copied
 - Snapshot notes the list of blocks

- Snapshots can also be deleted



Enabling and Taking a Snapshot

- Enable an HDFS directory for snapshotting in Cloudera Manager's HDFS File Browser tab

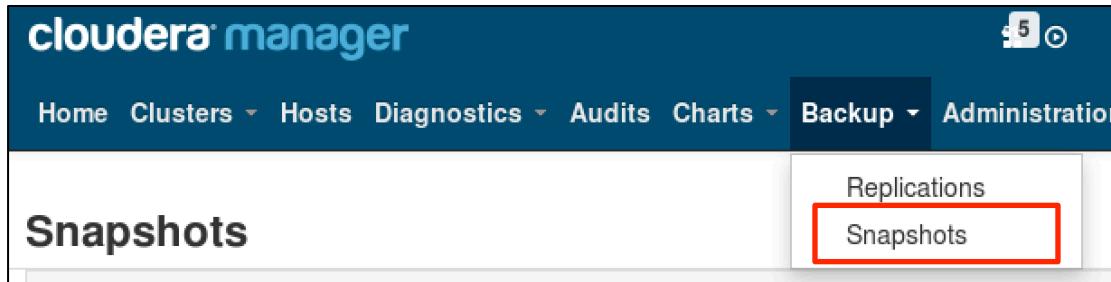
- Once snapshotting is enabled you can take a snapshot

The diagram illustrates the steps to enable and take a snapshot of an HDFS directory:

- Step 1: Enabling Snapshots**
The top panel shows the Cloudera Manager HDFS File Browser interface for the directory `/user/sample`. A red box highlights the `Enable Snapshots` button under the **Snapshots** section, which is currently disabled.
- Step 2: Taking a Snapshot**
The middle panel shows the **Snapshots** page for the same directory. A red box highlights the `Take Snapshot` button. Below it, a message indicates there are no snapshots yet.
- Step 3: Viewing the Snapshot**
The bottom panel shows the **Snapshots Show All** page, displaying a single snapshot entry: `March 31 2015 10:05 AM snapshot1`.

Snapshot Policies (1)

- **Cloudera Manager allows you to create snapshot policies**
 - To manage snapshot policies go to Backup > Snapshots



- **Snapshot policies define:**
 - HDFS directories to be snapshotted
 - Intervals at which snapshots should be taken
 - Number of snapshots to keep for each snapshot interval
- **Example:**
 - Create a policy that takes daily and weekly snapshots, and specify that 7 daily snapshots and 4 weekly snapshots should be maintained

Snapshot Policies (2)

- **Option to configure alerts on snapshot attempts**
 - For example: send an alert of the snapshot attempt failed
- **Managing snapshots**
 - If the snapshot policy includes a limit on the number of snapshots to keep, CM deletes older snapshots as needed
 - If you edit or delete a snapshot policy
 - Files or directories previously included in the policy may leave “orphaned” snapshots
 - These must be deleted manually
- **Tip: avoid orphaned snapshots by deleting these snapshots *before* editing or deleting the associated snapshot policy**

Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- **Hands-On Exercise: Taking HDFS Snapshots**
- Cluster Upgrading
- Conclusion

Hands-On Exercise: Directory Snapshots

- In this Hands-On Exercise, you will enable and take an HDFS snapshot, then delete and restore data from the snapshot
- Please refer to the Hands-On Exercise Manual

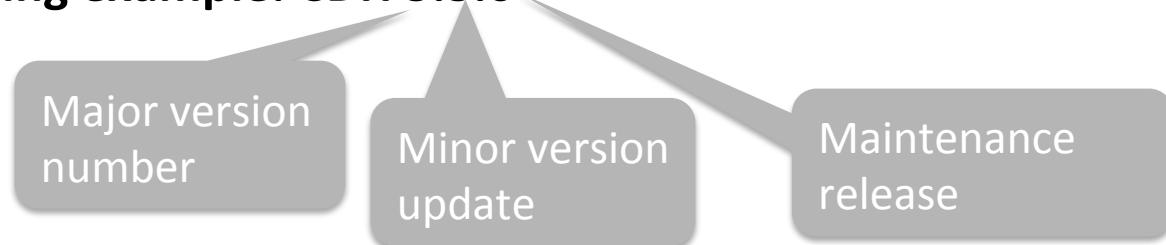
Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- **Cluster Upgrading**
- Conclusion

Upgrading Software: When to Upgrade?

- **Software numbering example: CDH 5.3.0**



- **Cloudera updates CDH regularly**

- Major versions: every 12 to 18 months
- Minor versions: every three to four months
- Maintenance releases: when necessary

- **We recommend that you upgrade when a new version update is released**

- **Cloudera supports:**

- CDH major versions for at least three years after the initial release (starting with CDH 4.0)
- CM major and minor versions for 1 year after GA of follow on release

Upgrading Software: General Notes

- **Software upgrade procedure is fully documented on the Cloudera Web site**
- **Cloudera Manager 5 supports clusters running CDH 4 or CDH 5**
 - Cloudera Manager minor version must be \geq CDH minor version
 - Example: to upgrade to CDH 5.2.1, CM 5.2.0 or later required
 - Parcel and Package installations are supported
 - Parcels installed by Cloudera Manager
 - Package installation is manual
 - Cloudera Manager 5.3 introduced an enhanced CDH upgrade wizard

Maintenance Release Upgrade – General Procedures (1)

- **Upgrading to a new maintenance release**
 - For example, from CDH 5.2.0 to CDH 5.2.1
- **Before Upgrading CDH – general procedure:**
 1. Run the Host Inspector (fix any issues)
 2. Run the Security Inspector (fix any issues)
 3. Run `hdfs fsck /` and `hdfs dfsadmin -report` (fix any issues)
 4. Reserve a maintenance window
 5. Enable maintenance mode before starting the upgrade
 - Avoids alerts during the upgrade
 - Remember to exit maintenance mode when upgrade completed
- **Run the ‘Upgrade Cluster’ wizard**

Maintenance Release Upgrade – General Procedures (2)

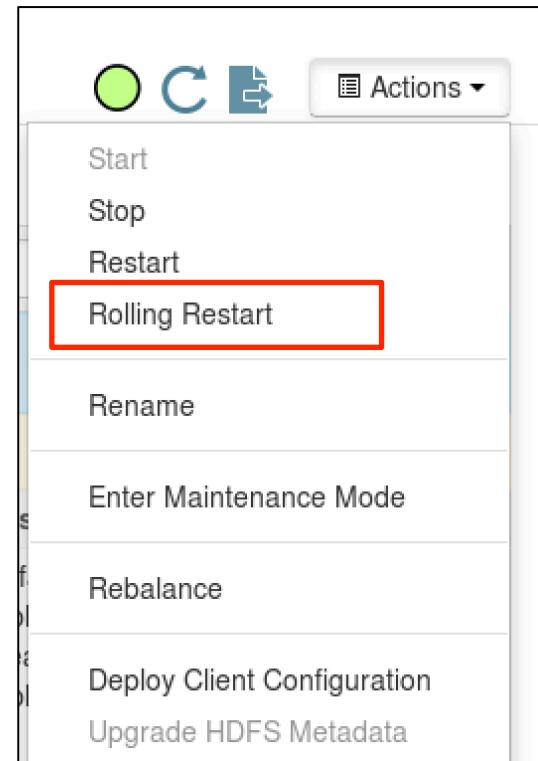
- **If using Parcels, the wizard *automatically* completes the necessary steps...**
 1. Confirms parcel availability
 2. Downloads and distributes parcel to nodes
 3. Shuts down services
 4. Activates new parcel
 5. Upgrades services as necessary
 6. Deploys client configuration files
 7. Restarts services
 8. Runs the Host inspector
 9. Reports results of the upgrade
- **If using Packages:**
 - Manually create the needed repository file pointing to the CDH software
 - Manually install needed CDH packages (using yum, apt-get, or zypper)
 - Run the upgrade wizard in Cloudera Manager

Minor Release Upgrade – General Procedures

- **Minor release upgrade**
 - For example, from CDH 5.2 to CDH 5.3
- **Same as for a maintenance release, but with some additional steps:**
 - After enabling Maintenance mode:
 - Stop Cluster Services
 - Back up the NameNode's HDFS metadata
 - After running the upgrade wizard:
 - If using Packages, remove old CDH version packages
 - Finalize the HDFS metadata upgrade (button on Cloudera Manager's NameNode page)
- **The above provides a general overview of the software upgrade process**
 - Consult the CDH technical documentation for details for upgrading to a specific release!

Rolling Upgrades with Cloudera Manager Enterprise Edition

- CM Enterprise Edition allows you to upgrade and restart upgraded services, with no downtime
- Requirements
 - Cloudera Manager *Enterprise* Edition
 - Parcels installation approach
 - HDFS High Availability
 - Major version upgrades (e.g., CDH 4 to CDH 5) are not supported
- General procedure:
 - Download, distribute, and activate new parcel
 - Rolling restart individual services or the entire cluster



Chapter Topics

Cluster Maintenance

- Checking HDFS Status
- Hands-On Exercise: Breaking The Cluster
- Copying Data Between Clusters
- Adding and Removing Cluster Nodes
- Rebalancing the Cluster
- Hands-On Exercise: Verifying The Cluster's Self-Healing Features
- Directory Snapshots
- Hands-On Exercise: Taking HDFS Snapshots
- Cluster Upgrading
- Conclusion

Essential Points

- You can check the status of HDFS with the `hdfs fsck` command
 - Reports problems but does not repair them
- You can use the `distcp` command to copy data within a cluster or between clusters
- Add hosts to or remove hosts from a cluster using Cloudera Manager
 - Specify rack when adding hosts
- Rebalance after adding new DataNodes to adjust block placement across HDFS, so as to ensure better utilization
- Cloudera Manager and the use of parcels help automate many software upgrade steps



Cluster Monitoring and Troubleshooting

Chapter 15



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting**
- Conclusion

Cluster Monitoring and Troubleshooting

In this chapter, you will learn:

- **What tools Cloudera Manager provides for monitoring**
- **Recommended items to monitor on a Hadoop cluster**
- **Some techniques for troubleshooting problems on a Hadoop cluster**
- **Some common misconfigurations, and their resolutions**

Monitoring Hadoop Clusters

- Use of a monitoring tool is important to warn you of potential or actual problems on individual machines in the cluster
- You can integrate cluster monitoring into many existing monitoring tools using JMX broadcasts and Metrics sinks
 - For example, Nagios or Ganglia
 - This is a specialized topic. Consult the Cloudera documentation for details
- Cloudera Manager provides comprehensive cluster monitoring with no additional configuration required
- This chapter first introduces the monitoring features provided by Cloudera Manager and then provides monitoring recommendations

Chapter Topics

Cluster Monitoring and Troubleshooting

- **Cloudera Manager Monitoring Features**
- Hands-On Exercise: Configuring Email Alerts
- Monitoring Hadoop Clusters
- Troubleshooting Hadoop Clusters
- Common Misconfigurations
- Conclusion
- Hands-On Exercise: Troubleshooting Challenge

Monitoring with Cloudera Manager

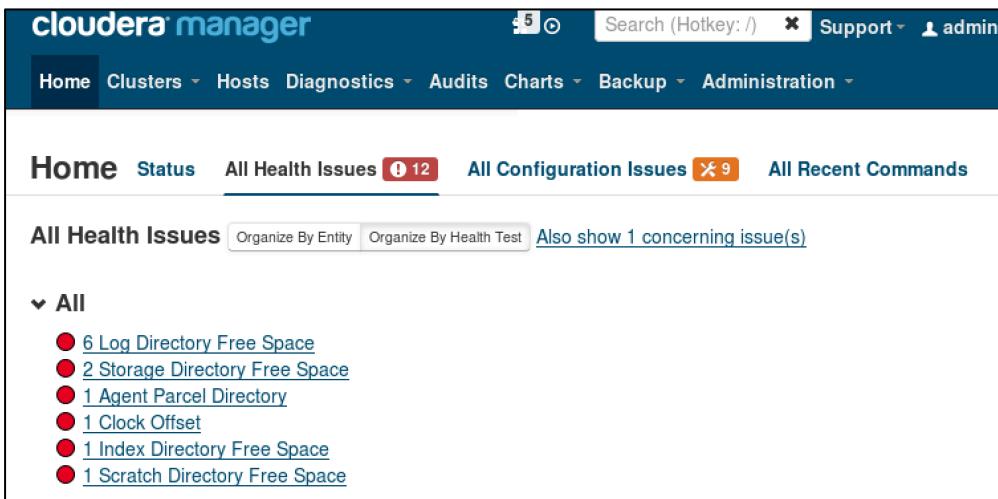
- **Cloudera Manager provides features to monitor the health and performance of clusters, services, hosts, and applications**
 - Monitor cluster health
 - Identify configuration issues
 - Track metrics and resource usage with charts and dashboards
 - View event logs
 - Generate alerts
 - View audits
 - Generate reports

Health Tests

- **Cloudera Manager monitors the health of services, roles, and hosts**
 - Examples:
 - Check if NameNode directories have enough disk space
 - Verify all DataNodes are connected to a NameNode
- **Health checks have threshold settings**
 - Example: set a threshold for permissible percentage of DataNodes down
 - Test can be pass-fail or metric-based
 - Health checks are enabled by default with standard thresholds
- **Health test indicators in Cloudera Manager**
 -  Good – test result above all thresholds
 -  Concerning – test result below the ‘warning’ threshold
 -  Bad – test result below the ‘critical’ threshold

Discovering Health Issue Details

- When you see this  icon, it indicates that the service has at least one health issue
 - The indicator color will match the worst level discovered
 - Click the indicator to display the specific health issues
- The Home page has links for ‘All Health Issues’
 - Option to organize view of all issues by Entity or by Health Test
 - Links to details of health test results

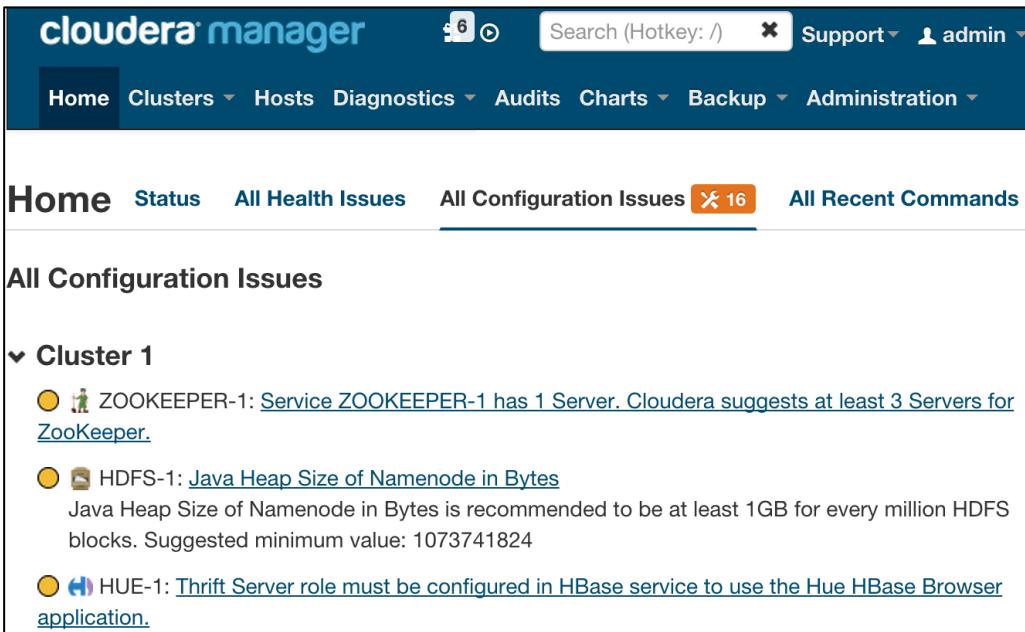


The screenshot shows the Cloudera Manager Home page. At the top, there's a navigation bar with links for Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. Below the navigation bar, there are several status indicators: Home (green), Status (green), All Health Issues (red, with a count of 12), All Configuration Issues (orange, with a count of 9), and All Recent Commands (blue). The 'All Health Issues' section is currently selected and expanded. It contains a link to 'Organize By Entity' and another to 'Organize By Health Test'. Below these links, a message says 'Also show 1 concerning issue(s)'. A list of 12 health issues is displayed, each preceded by a red circular icon with a white exclamation mark:

- 6 Log Directory Free Space
- 2 Storage Directory Free Space
- 1 Agent Parcel Directory
- 1 Clock Offset
- 1 Index Directory Free Space
- 1 Scratch Directory Free Space

Configuration Issues

- This icon  indicates that the specified number of configuration issues were found for the service
 - Click the icon to display details
- The Home page provides a link to the 'All Configuration Issues' page
 - The page shows issue details and links to relevant configuration pages



The screenshot shows the Cloudera Manager interface. At the top, there's a navigation bar with 'cloudera manager' logo, a notification badge (6), a search bar ('Search (Hotkey: /)'), a support link, and a user account ('admin'). Below the header is a secondary navigation bar with links: Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration.

The main content area has a breadcrumb navigation: Home > All Configuration Issues. The 'All Configuration Issues' link is highlighted with a blue underline and an orange exclamation mark icon with the number '16'. Below this, the title 'All Configuration Issues' is displayed. A section titled 'Cluster 1' is expanded, showing three configuration issues:

- 🏛 ZOOKEEPER-1: [Service ZOOKEEPER-1 has 1 Server. Cloudera suggests at least 3 Servers for ZooKeeper.](#)
- 📁 HDFS-1: [Java Heap Size of Namenode in Bytes](#)
Java Heap Size of Namenode in Bytes is recommended to be at least 1GB for every million HDFS blocks. Suggested minimum value: 1073741824
- 🎙 HUE-1: [Thrift Server role must be configured in HBase service to use the Hue HBase Browser application.](#)

Time Range Selector

The screenshot shows the Cloudera Manager interface with the 'Hosts' tab selected. A red arrow points to a small diamond-shaped 'Time Marker' on the timeline, labeled 'Time Marker'. Another red arrow points to a text input field containing '2.7 days containing February 9 2015, 6:00 PM PST', labeled 'Click the Range Selector to reveal the calendar'. A third red arrow points to a magnifying glass icon, labeled 'Zoom in or out'. A fourth red arrow points to a 'Toggle timeline view on/off' button. Below the timeline, a modal window titled 'From (PST)' and 'To (PST)' is displayed, showing date pickers for both ends of the range, and dropdown menus for 'Time', 'Hour', and 'Minute' settings.

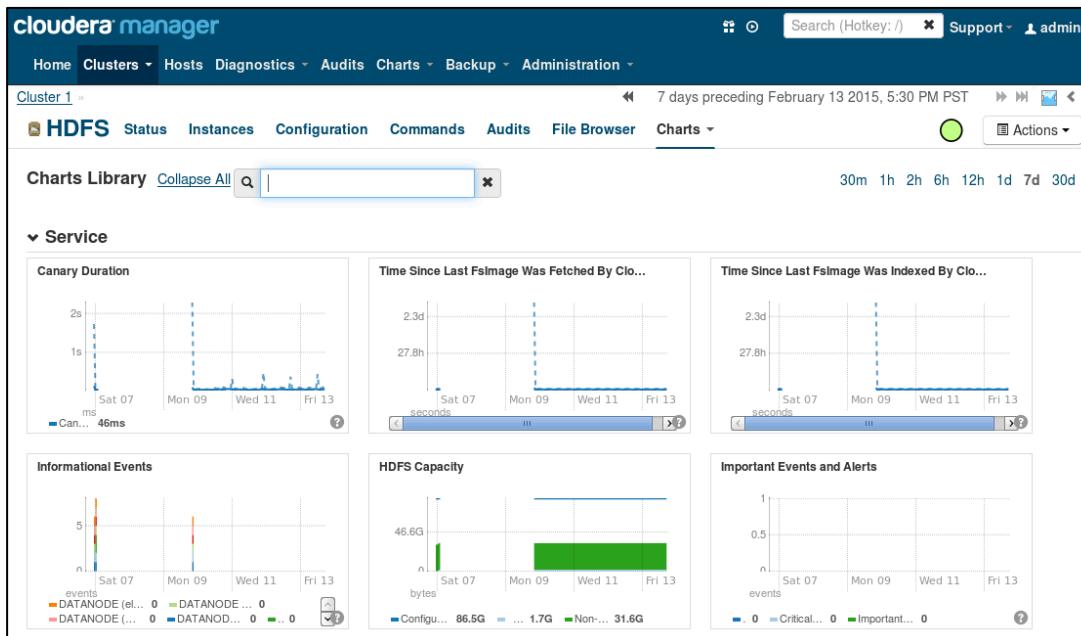
- Configure a page to show information for a specific time period
- The Time Marker allows you to select a specific point in time
 - For example on a service's 'Status' page
- Select a time range with the Range Selector
 - Available in many pages

Monitoring Terminology

- **Metric – a property that can be measured**
 - Cloudera Manager monitors performance metrics for services and role instances running on clusters
 - Examples: RAM utilization, total HDFS storage capacity
- **Entity – a Cloudera Manager component with metrics associated with it**
 - Examples: a service, a role, a host
- **Charts - display metrics for entities**
 - Metrics aggregated and accessible through charts
 - Custom charts can be created
- **Dashboard – a set of charts**
 - Custom dashboards can be configured and managed

Pre-Built Charts

- ‘Status’ pages in Cloudera Manager already include dashboards with some charts
 - You may optionally add charts on any of these pages
- Cloudera Manager ships with an extensive library of pre-built charts
 - Example: browse to HDFS > Charts > Library



Custom Charts

- Create custom charts to add to dashboards
- Custom charts are created with the tsquery language
- tsquery syntax
 - SELECT [metric expression] WHERE [predicate]
- Example
 - SELECT bytes_read_rate, bytes_written_rate WHERE roleType=DATANODE AND serviceName=HDFS

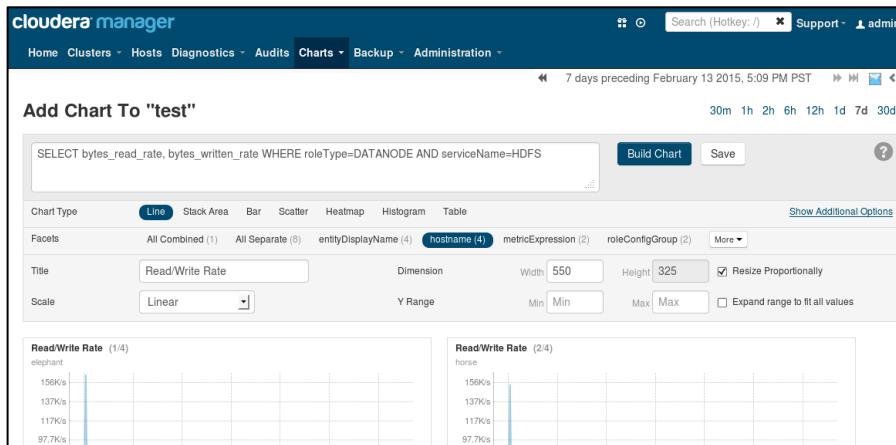
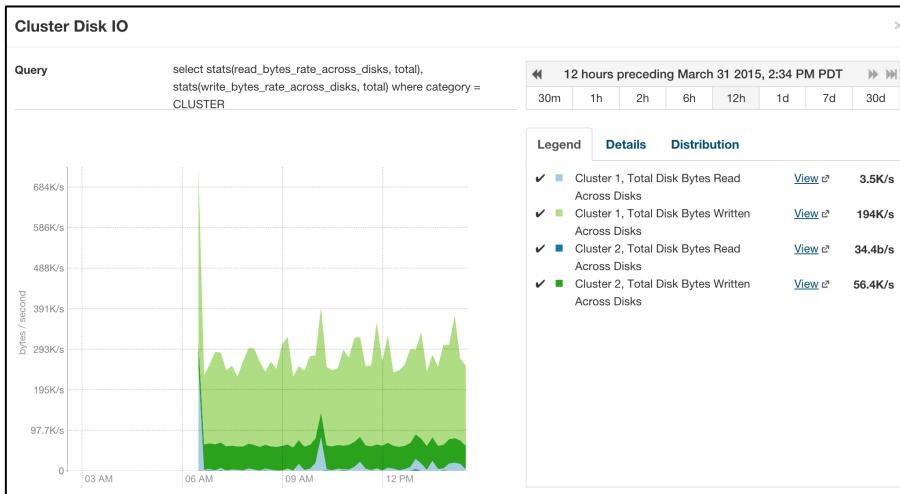


Chart Options

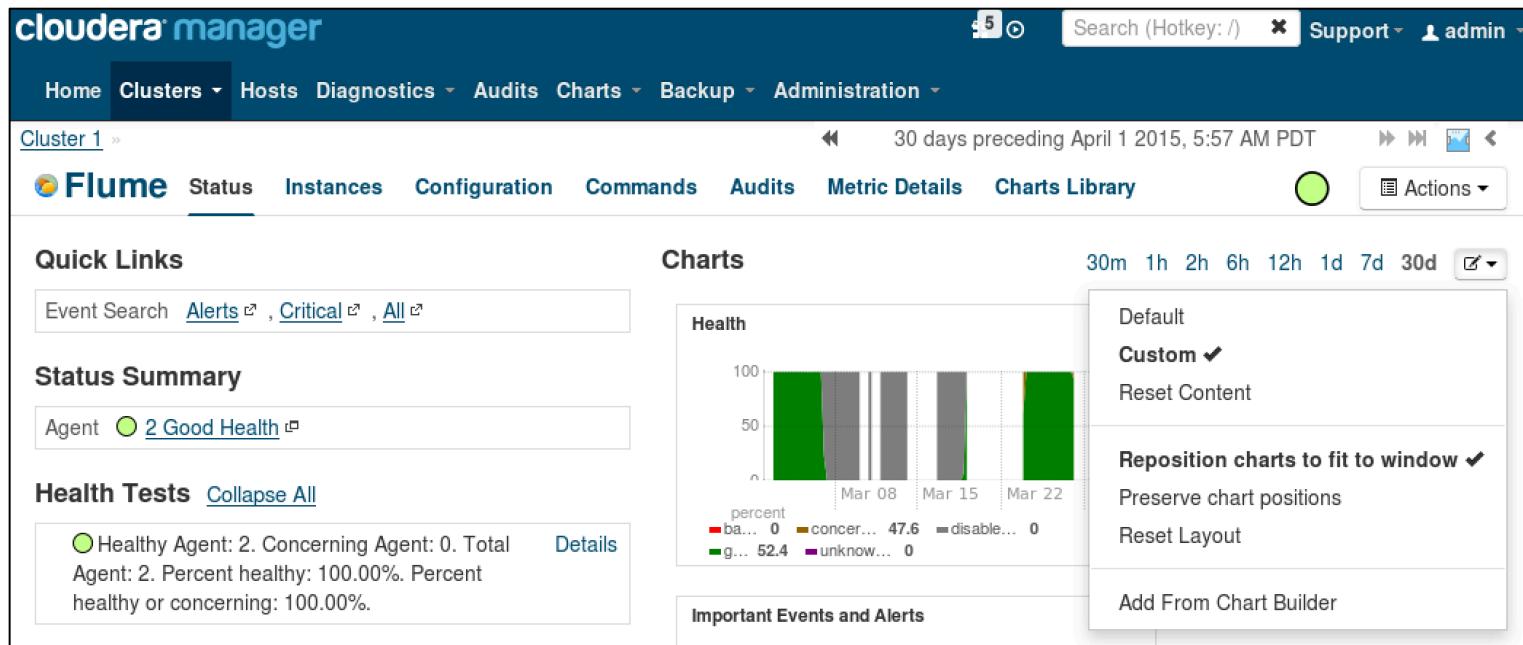
▪ Hover over a chart to see options

- Click on the  icon to expand the chart viewing area and details
- Click on the  icon for more options
 - Remove from the current page or add the chart to a dashboard
 - Export chart data in CSV or JSON format
 - Edit the chart
 - Create a trigger



Dashboards (1)

- Dashboards are sets of charts
- Many dashboards already exist
 - For example, on each service's Status page
 - Click on the  icon to manage an existing dashboard
 - Options to add or remove charts, change layout



The screenshot shows the Cloudera Manager interface for the Flume service. The top navigation bar includes Home, Clusters, Hosts, Diagnostics, Audits, Charts, Backup, and Administration. The sub-navigation for the Flume cluster shows Status, Instances, Configuration, Commands, Audits, Metric Details, and Charts Library. A green circular status indicator and an Actions button are also present.

Quick Links: Event Search (Alerts, Critical, All).

Status Summary: Agent 2 Good Health.

Health Tests: Healthy Agent: 2. Concerning Agent: 0. Total Agent: 2. Percent healthy: 100.00%. Percent healthy or concerning: 100.00%.

Charts: A chart titled "Health" displays the percentage of healthy agents over time. The legend indicates:

- ba... 0 (red)
- concer... 47.6 (orange)
- disable... 0 (grey)
- g... 52.4 (green)
- unknow... 0 (purple)

Approximate data points: Mar 08 (~47.6%), Mar 15 (~52.4%), Mar 22 (~52.4%).

Actions Menu: A dropdown menu on the right provides options for managing the dashboard layout:

- Default
- Custom (selected)
- Reset Content
- Reposition charts to fit to window (selected)
- Preserve chart positions
- Reset Layout
- Add From Chart Builder

Dashboards (2)

- Create a ‘Custom Dashboard’ by choosing Charts > New Dashboard
- Add existing or custom charts to a dashboard
 - Example: browse a service’s Charts Library, select a chart and add it to the dashboard

The screenshot shows the Cloudera Manager interface for managing dashboards. At the top, there's a navigation bar with tabs: Cluster 1, Flume, Status, Instances, Configuration, Commands, Audits, Metric Details, and Charts Library. The Charts Library tab is selected. Below the navigation bar, there's a search bar and a 'Save Chart' button. On the left, there's a sidebar labeled 'Service' with a dropdown menu. The main area displays two charts: 'Informational Events' and 'Important Events and Alerts'. The 'Informational Events' chart has a red box around its 'Add to Dashboard' button, which is highlighted by a red arrow pointing to the 'Save Chart' dialog box on the right. The 'Save Chart' dialog box has 'Informational Events' entered in the 'Chart Title' field. It contains two radio button options: 'Add chart to an existing custom or system dashboard' (selected) and 'Add chart to a new custom dashboard'. The 'Dashboard Name' dropdown is set to 'Select a Dashboard', which is also highlighted by a red arrow. A dropdown menu is open, showing 'Custom Dashboards' (selected), 'My Dashboard', and 'System Dashboards' (with sub-options: CDH5 Flume Agent Status Page, CDH5 Flume Status Page, CDH5 HDFS DataNode Status Page).

- Manage your user-defined dashboards in Charts > Manage Dashboards

Events

- An ‘event’ is a record of something of interest that occurred
 - Default settings enable the capture of many events
- Any of the following can be captured as events:
 - Jobs that fail or run slowly
 - Actions taken in Cloudera Manager such as starting a role
 - Health test results
 - Log messages from HDFS, HBase, or MapReduce

■ How to view events

- Go to Diagnostics > Events
- Set a time frame
- Add filters (optional)
- Click Search

■ Events can be used for alerting

The screenshot shows the Cloudera Manager interface with the 'Diagnostics' tab selected. The main area is titled 'Events' and displays two log entries:

Date	Level	Message	Action
March 31 2015 5:30 PM	CRITICAL	Command HdfsScheduledSnapshotsCommand on subject hdfs succeeded.	View ↗
March 31 2015 5:30 PM	INFORMATIONAL	User clouderaManager ran a scheduled snapshots management command on HDFS service hdfs.	View ↗

Alerts

- **Manage alerts in Cloudera Manager under Administration > Alerts**
 - Some alerts are enabled by default
- **Alerts can be configured for any of the following conditions:**
 - Activity running too slowly
 - A configuration was changed
 - Health condition thresholds not met on a role or host
 - Log messages that match a condition you define
- **Configure alert delivery**
 - Send alerts as emails to addresses you specify
 - Specify email server during CM installation
 - Also configurable in Cloudera Management Service's Configuration
 - Alerts can also be sent as SNMP traps

Where to Configure Alerting

- Administration > Alerts page
- Properties in each service's configuration tab
 - Rules to Extract Events from Log Files
 - Enable Health Alerts for this Role

DataNode Group 1 / Monitoring

Rules to Extract Events from Log Files

```
version": "0",
"rules": [
  {"alert": false, "rate": 1, "periodminutes": 1, "threshold":"FATAL"},
  {"alert": false, "rate": 0, "threshold":"WARN", "content": ". is deprecated. Instead, use \"."},
  {"alert": false, "rate": 0, "threshold":"WARN", "content": ". is deprecated. Use \"." instead"},
  {"alert": false, "rate": 0, "exceptiontype": "java.io.IOException"},
  {"alert": false, "rate": 0, "exceptiontype": "java.net.SocketException"},
  {"alert": false, "rate": 0, "exceptiontype": "java.net.SocketClosedException"},
  {"alert": false, "rate": 0, "exceptiontype": "java.io.EOFException"},
  {"alert": false, "rate": 0, "exceptiontype": "java.nio.channels.CancelledKeyException"},
  {"alert": false, "rate": 1, "periodminutes": 5, "content": "DataNode registration failed"},
  {"alert": false, "rate": 1, "periodminutes": 2, "exceptiontype": ".+"},
  {"alert": false, "rate": 0, "threshold": "WARN", "content": "Got a command from standby NN - ignoring command:.+"},
  {"alert": false, "rate": 0, "threshold": "WARN", "content": "Unknown job [^ ]+ being deleted."},
  {"alert": false, "rate": 0, "threshold": "WARN", "content": "Error executing shell command .+ No such process.+"},
  {"alert": false, "rate": 0, "threshold": "WARN", "content": ".+ attempt to override final parameter.+"},
  {"alert": false, "rate": 0, "threshold": "WARN", "content": "[^ ]+ is a deprecated filesystem name. Use.\"."},
  {"alert": false, "rate": 1, "periodminutes": 1, "threshold": "WARN"}
]
```

Reset to the default value:

cloudera manager

Home Clusters Hosts Diagnostics Audits Charts Backup Administration

Alerts

Recipient(s): [training@lion](#) Send Test Alert

Alert On Transitions Out of Alerting Health: No

Health Alert Threshold: Bad

Alert Type	Health Alert Settings
Health	Hosts (<input checked="" type="checkbox"/>) Alerts Enabled For: None
Hosts	Alerts Disabled For: All hosts
flume	flume (<input checked="" type="checkbox"/>)
hdfs	Alerts Enabled For: Agent health Service health
hive	Alerts Disabled For: None
hue	hdfs (<input checked="" type="checkbox"/>)
impala	
momt	

HIVE-1 Status Instances Configuration Commands Audits Charts

Configuration

enable health alerts role

Category	Property	Value
Hive Metastore Server Default Group / Monitoring	Enable Health Alerts for this Role	<input checked="" type="checkbox"/> default value
HiveServer2 Default Group / Monitoring	Enable Health Alerts for this Role	<input checked="" type="checkbox"/> default value

Audits

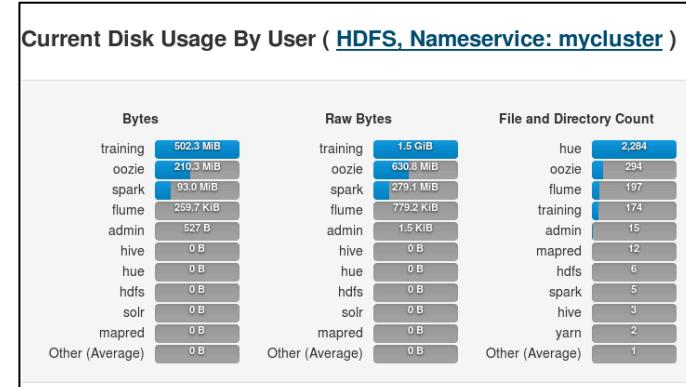
- Audit events show actions that occurred on a host or for a service or role
- In the Audits tab of Cloudera Manager
 - Set a time frame
 - Apply one or more filters (optional)
 - Click Search and view the audit event log results
 - Results also downloadable

The screenshot shows the Cloudera Manager interface with the 'Audits' tab selected. A blue bracket on the left points to the 'User-defined filters' section, which includes dropdown menus for 'Service' (set to 'HDFS') and 'Command' (set to 'RESTART'), along with a search button and a 'Download CSV' link. Above the filters, a time range is set from '02 PM' to '04:45'. Below the filters, two audit log entries are listed:

Date	Action	Details
March 31 2015 12:49 PM	RESTART	Succeeded command Restart on service hdfs Service: hdfs
March 31 2015 12:48 PM	RESTART	Started command Restart on service hdfs User: admin IP Address: 172.30.0.67 Service: hdfs

Reports

- Reports are a feature of Cloudera Enterprise
- Access reports from the Clusters menu in Cloudera Manager
- Reports are available for disk usage, YARN applications, Impala queries, and HBase tables and namespaces
 - Download reports or view them in Cloudera Manager
 - Reports are per-cluster
- You can also generate HDFS custom reports



cloudera manager

Home Clusters ▾ Hosts Diagnostics ▾ Audits Charts ▾ Backup ▾ Administration ▾

Cluster 1

Reports

Disk Usage ([HDFS](#))

Title

Nameservice: mycluster

[Current Disk Usage By User](#) Download CSV XLS

[Current Disk Usage By Group](#) CSV XLS

[Current Disk Usage By Directory](#) CSV XLS

[Historical Disk Usage By User](#) CSV XLS

[Historical Disk Usage By Group](#) CSV XLS

[Historical Disk Usage By Directory](#) CSV XLS

[Large Files](#)

[Large Directories](#)

[Files with Low Replication](#)

Chapter Topics

Cluster Monitoring and Troubleshooting

- Cloudera Manager Monitoring Features
- **Hands-On Exercise: Configuring Email Alerts**
- Monitoring Hadoop Clusters
- Troubleshooting Hadoop Clusters
- Common Misconfigurations
- Conclusion
- Hands-On Exercise: Troubleshooting Challenge

Hands-On Exercise: Configuring Email Alerts

- In this Hands-On Exercise, you will configure Cloudera Manager to use an email server to send alerts.
- Please refer to the Hands-On Exercise Manual

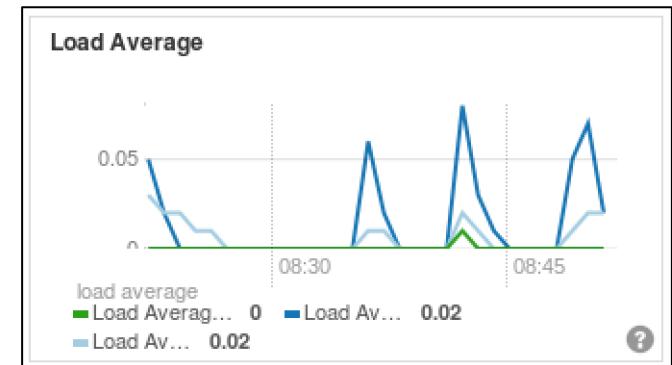
Chapter Topics

Cluster Monitoring and Troubleshooting

- Cloudera Manager Monitoring Features
- Hands-On Exercise: Configuring Email Alerts
- **Monitoring Hadoop Clusters**
- Troubleshooting Hadoop Clusters
- Common Misconfigurations
- Conclusion
- Hands-On Exercise: Troubleshooting Challenge

Items to Monitor (1)

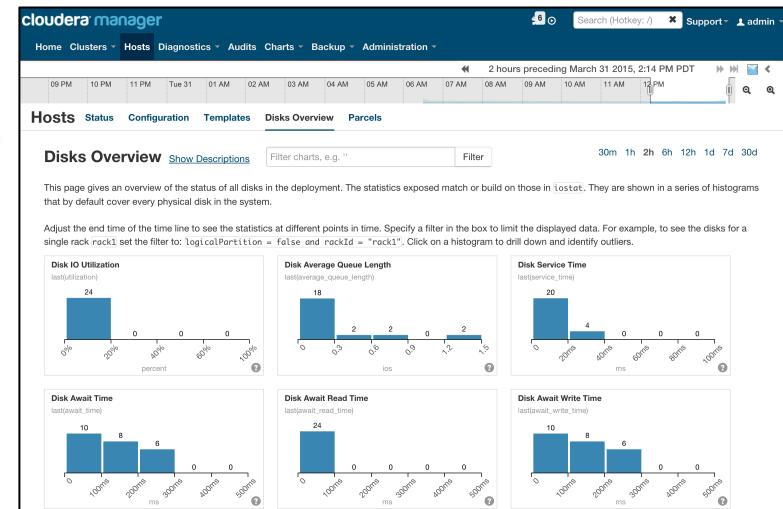
- **Monitor the Hadoop daemons**
 - See status of daemons in service pages of Cloudera Manager
 - Recommendation: send an alert if a daemon goes down
 - CM default will alert for some but not all daemons down
- **Monitor CPU usage on master nodes**
 - Browse to Host page of each master node for “Host CPU Usage”, “Roles CPU Usage” and “Load Average” charts
 - Recommendation: Track excessive CPU usage and load averages for performance and utilization purposes
 - Worker nodes will often reach 100% usage
 - This is not a problem



Items to Monitor (2)

■ Monitor disks and disk partitions

- Hosts page ‘Disk Overview’ tab →
- Also, from a Cluster page, choose Configuration > All Disk Space Thresholds



■ Recommendations:

- Alert immediately if a disk fails
 - Controlled by ‘DataNode Volume Failures Thresholds’ property (CM Default: any disk failure is ‘critical’)
 - Enable health test alerts for the DataNode role (CM default: disabled)
- Send a *warning* when a master node disk reaches 80% capacity and a *critical alert* when a disk reaches 90% capacity
 - Controlled by ‘DataNode Free Space Monitoring Thresholds’ property

Items to Monitor (3)

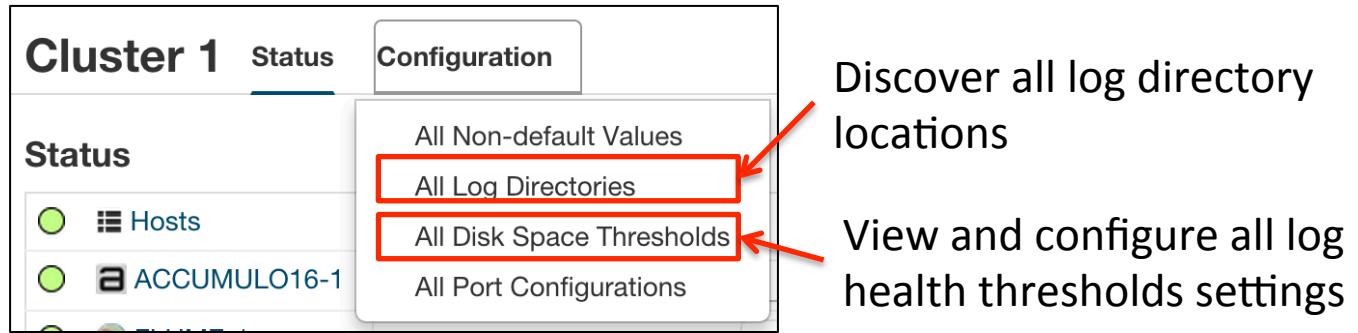
- **Monitor swap on all nodes**
 - Monitored by ‘Host Memory Swapping Thresholds’ property
 - Alert if the swap partition starts to be used
 - Memory allocation is overcommitted
- **Monitor network transfer speeds**
- **Monitor HDFS health**
 - HA configuration
 - Check the size of the edit logs on the JournalNodes
 - Monitor for failovers
 - Non-HA configuration
 - Check the age of the `fsimage` file and/or check the size of the `edits` file

Log File Growth (1)

- **HDFS and YARN/MRv2 daemon logs**
 - HDFS and YARN/MRv2 daemons log use Log4j's Rolling File Appender, so logs are rotated
 - Configure appropriate size and retention policies in Cloudera Manager
- **YARN Application logs**
 - Caution: inexperienced developers will often create large container logs from their applications
 - Large task logs impact disk usage
 - The log files are originally written to local disk – *not* HDFS – on the worker nodes
 - With log file aggregation, task logs get moved to HDFS
 - Without log file aggregation, task logs stay on local disk

Log File Growth (2)

- Ensure you have enough room for YARN application container logs
- Monitor local disk and HDFS to ensure that developers are not logging excessively



- Configure properties to control task log growth
 - `yarn.nodemanager.log-retain-seconds`
 - Retention of local log files when log aggregation is not enabled
 - CM Default: 3 hours
 - `yarn.log-aggregation.retain-seconds`
 - Retention of log files on HDFS when log aggregation is enabled
 - CM Default: 7 days

Chapter Topics

Cluster Monitoring and Troubleshooting

- Cloudera Manager Monitoring Features
- Hands-On Exercise: Configuring Email Alerts
- Monitoring Hadoop Clusters
- **Troubleshooting Hadoop Clusters**
- Common Misconfigurations
- Conclusion
- Hands-On Exercise: Troubleshooting Challenge

Troubleshooting: The Challenges

- **An overt symptom is a poor predictor of the root cause of a failure**
- **Errors show up far from the cause of the problem**
- **Clusters have a lot of components**
- **Example:**
 - Symptom: A Hadoop job that previously was able to run now fails to run
 - Cause: Disk space on many nodes has filled up, so intermediate data cannot be copied to Reducers

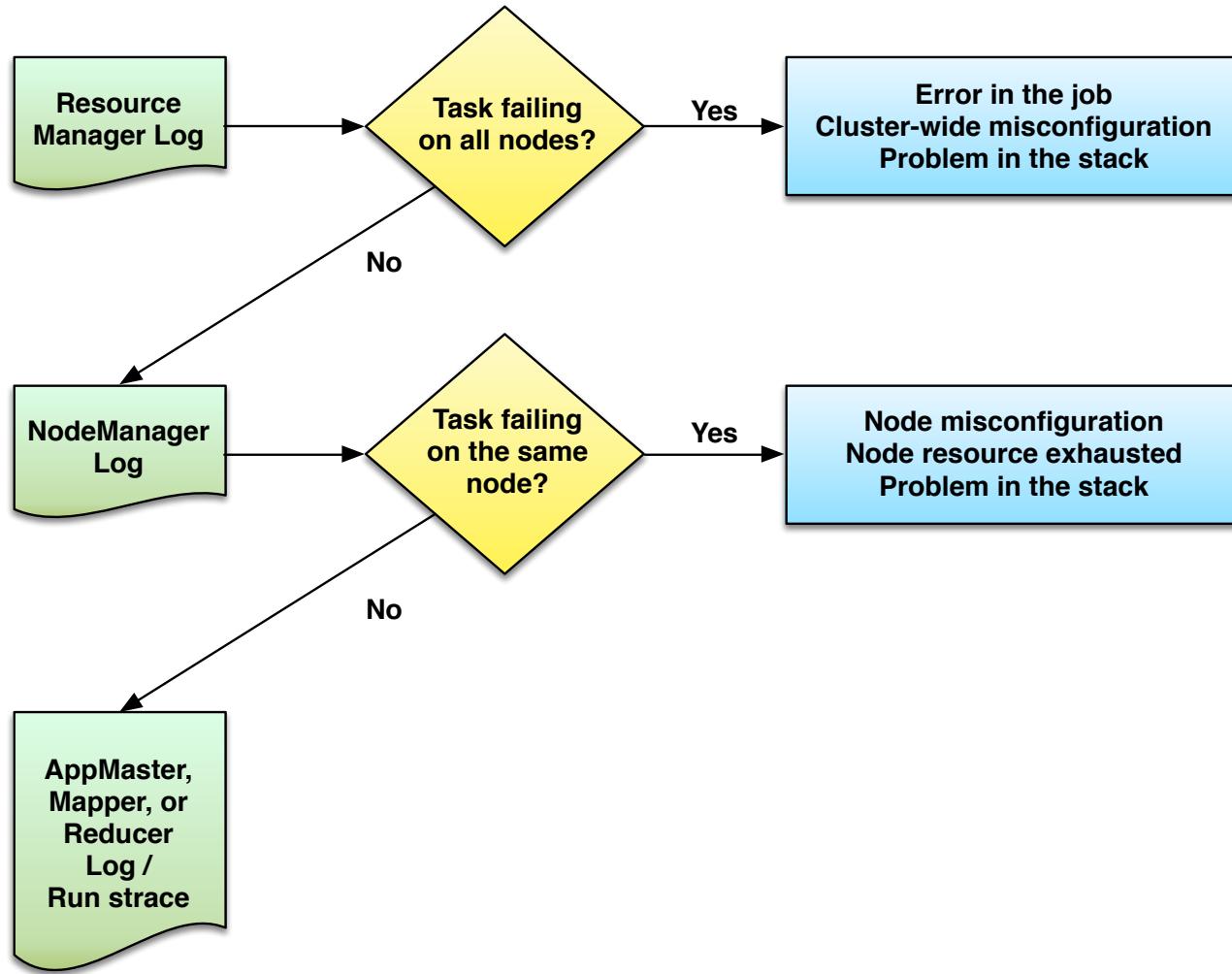
Common Sources of Problems

- **Misconfiguration**
- **Hardware failure**
- **Resource exhaustion**
 - Not enough disk
 - Not enough RAM
 - Not enough network bandwidth
- **Inability to reach hosts on the network**
 - Naming issues
 - Network hardware issues
 - Network delays

Gathering Information About Problems

- **Are there any issues in the environment?**
- **What about dependent components?**
 - YARN applications depend on the ResourceManager, which depends on the underlying OS
- **Is there any predictability to the failures?**
 - All from the same application?
 - All from the same NodeManager?
- **Is this a resource problem?**
 - Have you received an alert from Cloudera Manager?
- **What do the logs say?**
- **What does the CDH documentation/the Cloudera Knowledge Base/your favorite Web search engine say about the problem?**

General Rule: Start Broad, Then Narrow the Scope



Avoiding Problems to Begin With

■ Misconfiguration

- Start with recommended configuration values
 - Don't rely on Hadoop's defaults!
- Understand the precedence of overrides
- Control your clients' ability to make configuration changes
- Test changes before putting them into production
- Look for changes when deploying new releases of Hadoop
- Automate management of the configuration

■ Hardware failure and exhaustion

- Monitor your systems
- Benchmark systems to understand their impact on your cluster

■ Hostname resolution

- Test forward and reverse DNS lookups

Chapter Topics

Cluster Monitoring and Troubleshooting

- Cloudera Manager Monitoring Features
- Hands-On Exercise: Configuring Email Alerts
- Monitoring Hadoop Clusters
- Troubleshooting Hadoop Clusters
- **Common Misconfigurations**
- Conclusion
- Hands-On Exercise: Troubleshooting Challenge

Common Misconfigurations: Introduction

- **35% of Cloudera support tickets are due to misconfigurations**
- **In this section, we will explore some of the most common Hadoop misconfigurations and suggested solutions**
- **Note that these are just some of the issues you could run in to on a cluster**
- **Also note that these are *possible* causes and resolutions**
 - The problems could be caused by many other issues

Map/Reduce Task Out Of Memory Error (1)

```
2015-03-11 18:51:13,810 FATAL [main] org.apache.hadoop.mapred.YarnChild: Error running child : java.lang.OutOfMemoryError: Java heap space at java.util.Arrays.copyOf((Arrays.java:2271) at java.io.ByteArrayOutputStream.grow(ByteArrayOutputStream.java:113) at java.io.ByteArrayOutputStream.ensureCapacity(ByteArrayOutputStream.java:93) at java.io.ByteArrayOutputStream.write(ByteArrayOutputStream.java:140) at java.io.OutputStream.write(OutputStream.java:75) at HOTMapper.setup(HOTMapper.java:48) at org.apache.hadoop.mapreduce.Mapper.run(Mapper.java:142) at org.apache.hadoop.mapred.MapTask.runNewMapper(MapTask.java:764) at org.apache.hadoop.mapred.MapTask.run(MapTask.java:340) at org.apache.hadoop.mapred.YarnChild$2.run(YarnChild.java:165) at java.security.AccessController.doPrivileged(Native Method) at javax.security.auth.Subject.doAs(Subject.java:415) at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1548) at org.apache.hadoop.mapred.YarnChild.main(YarnChild.java:160)
```

Map/Reduce Task Out Of Memory Error (2)

■ Symptom

- A task fails to run

■ Possible causes

- Poorly coded Mapper or Reducer
- Map or Reduce task has run out of memory
- There is a memory leak in the code

■ Possible resolution

- Increase size of RAM allocated in `mapreduce.map.java.opts` and/or `mapreduce.reduce.java.opts`
- Ensure `mapreduce.task.io.sort.mb` is smaller than RAM allocated in `mapreduce.map.java.opts`
- Require the developer to recode a poorly-written Mapper or Reducer

Not Able To Place Enough Replicas

```
WARN org.apache.hadoop.hdfs.server.namenode.FSNamesystem: Not able to place enough replicas
```

■ Symptom

- Inadequate replication or application failure

■ Possible causes

- DataNodes do not have enough transfer threads
- Fewer DataNodes available than the replication factor of the blocks

■ Possible resolutions

- Increase `dfs.datanode.max.transfer.threads`
- Check replication factor

Where Did My File Go?

```
$ hdfs dfs -rm -r data  
$ hdfs dfs -ls /user/training/.Trash
```

■ Symptom

- User cannot recover an accidentally deleted file from the trash

■ Possible causes

- Trash is not enabled
- Trash interval is set too low

■ Possible resolution

- Set `fs.trash.interval` to a higher value

Chapter Topics

Cluster Monitoring and Troubleshooting

- Cloudera Manager Monitoring Features
- Hands-On Exercise: Configuring Email Alerts
- Monitoring Hadoop Clusters
- Troubleshooting Hadoop Clusters
- Common Misconfigurations
- **Conclusion**
- Hands-On Exercise: Troubleshooting Challenge

Essential Points

- **Be sure to monitor your Hadoop cluster**
 - Hadoop daemons, disk usage, CPU usage, swap, network usage, and HDFS health
- **Cloudera Manager provides Hadoop cluster monitoring**
- **Troubleshooting Hadoop problems is a challenge, because symptoms do not always point to the source of problems**
- **Follow best practices for configuration management, benchmarking, and monitoring and you will avoid many problems**

Chapter Topics

Cluster Monitoring and Troubleshooting

- Cloudera Manager Monitoring Features
- Hands-On Exercise: Configuring Email Alerts
- Monitoring Hadoop Clusters
- Troubleshooting Hadoop Clusters
- Common Misconfigurations
- Conclusion
- **Hands-On Exercise: Troubleshooting Challenge**

Troubleshooting Challenge: Heap of Trouble

- In this Troubleshooting Challenge, you will recreate a problem scenario, diagnose the problem, and, if you have time, fix the problem
- Your instructor will provide direction as you go through the troubleshooting process
- Please refer to the Hands-On Exercise Manual



Conclusion

Chapter 16



Course Chapters

- Introduction
- The Case for Apache Hadoop
- Hadoop Cluster Installation
- The Hadoop Distributed File System (HDFS)
- MapReduce and Spark on YARN
- Hadoop Configuration and Daemon Logs
- Getting Data Into HDFS
- Planning Your Hadoop Cluster
- Installing and Configuring Hive, Impala, and Pig
- Hadoop Clients Including Hue
- Advanced Cluster Configuration
- Hadoop Security
- Managing Resources
- Cluster Maintenance
- Cluster Monitoring and Troubleshooting
- **Conclusion**

Conclusion (1)

During this course, you have learned:

- **The core technologies of Hadoop**
- **How to populate HDFS from external sources**
- **How to plan your Hadoop cluster hardware and software**
- **How to deploy a Hadoop cluster**
- **What issues to consider when installing Pig, Hive, and Impala**
- **What issues to consider when deploying Hadoop clients**
- **How Cloudera Manager can simplify Hadoop administration**
- **How to configure HDFS for high availability**
- **What issues to consider when implementing Hadoop security**

Conclusion (2)

- **How to schedule MapReduce jobs and Spark applications on the cluster**
- **How to maintain your cluster**
- **How to monitor, troubleshoot, and optimize the cluster**

Class Evaluation

- Please take a few minutes to complete the class evaluation
 - Your instructor will show you how to access the online form

Thank You!

- **Thank you for attending this course**
- **If you have any further questions or comments, please feel free to contact us**
 - Full contact details are on our Web site at
<http://www.cloudera.com/>