



**BITS** Pilani  
Pilani Campus

# BITS Pilani presentation

Dr. Vivek V. Jog  
Dept. Of Computer Engineering





# **Big Data Systems (S1-24\_CCZG522)**

## **Lecture No.6**

---

YET  
ANOTHER  
RESOURCE  
NEGOTIATOR

Coming Up.....



---

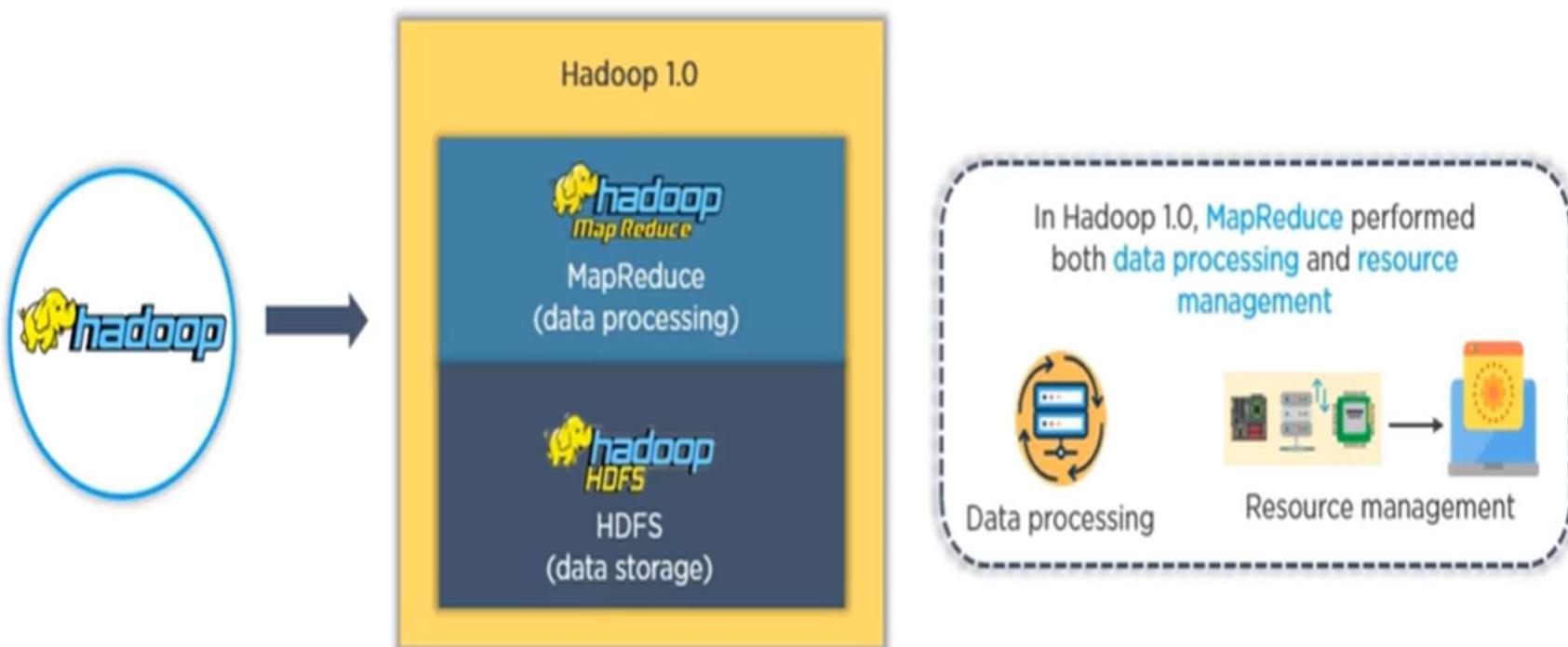
That is all for the day

Thank you

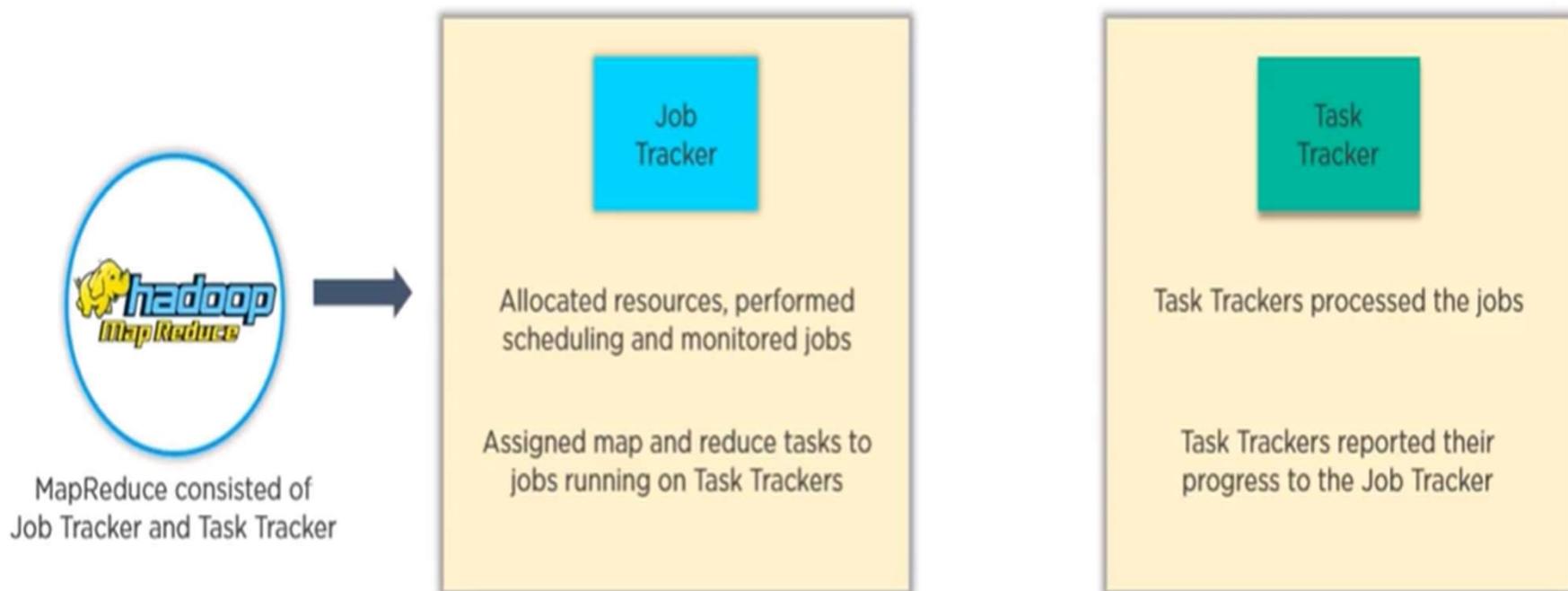
# Todays Agenda

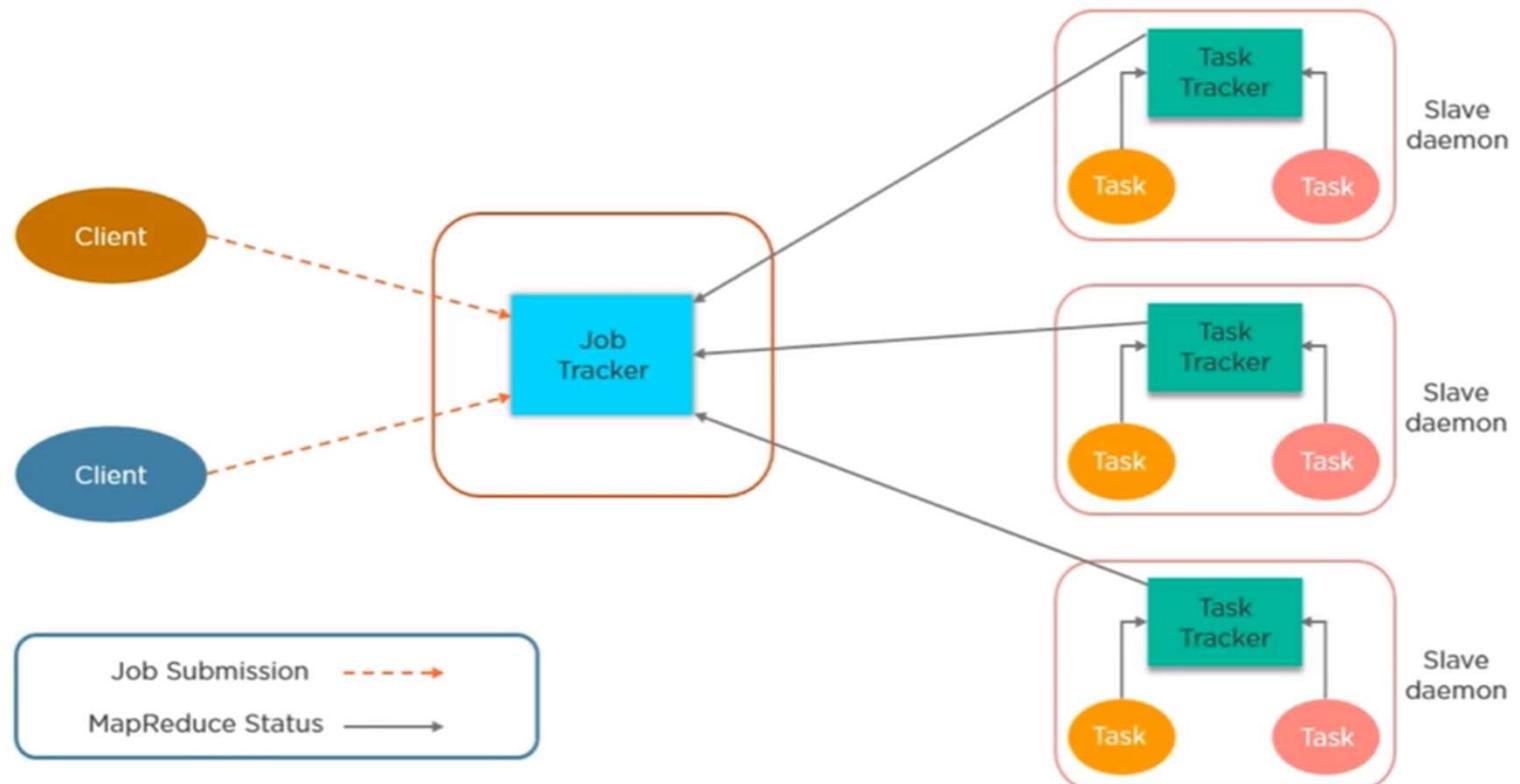


# Hadoop-1



# Hadoop-1





# Limitations

## 1 Scalability

Due to a **single JobTracker**, scalability became a bottleneck.  
Cannot have a cluster size of more than **4000 nodes** and cannot run more than **40000 concurrent tasks**

## 2 Availability issue

JobTracker is **single point of failure**. Any failure kills all queued and running jobs. Jobs need to be resubmitted by users

## 3 Resource Utilization

Due to predefined number of **map** and **reduce slots** for each TaskTracker, **resource utilization** issues occur

## 4 Limitations in running non-MapReduce applications

Problem in performing **real-time analysis** and running **Ad-hoc query** as MapReduce is batch driven

# Solution

Scalability



Can have a cluster size of more than 10,000 nodes and can run more than 1,00,000 concurrent tasks

Compatibility



Applications developed for Hadoop 1 runs on YARN without any disruption or availability issues

Resource utilization



Allows dynamic allocation of cluster resources to improve resource utilization

Multitenancy

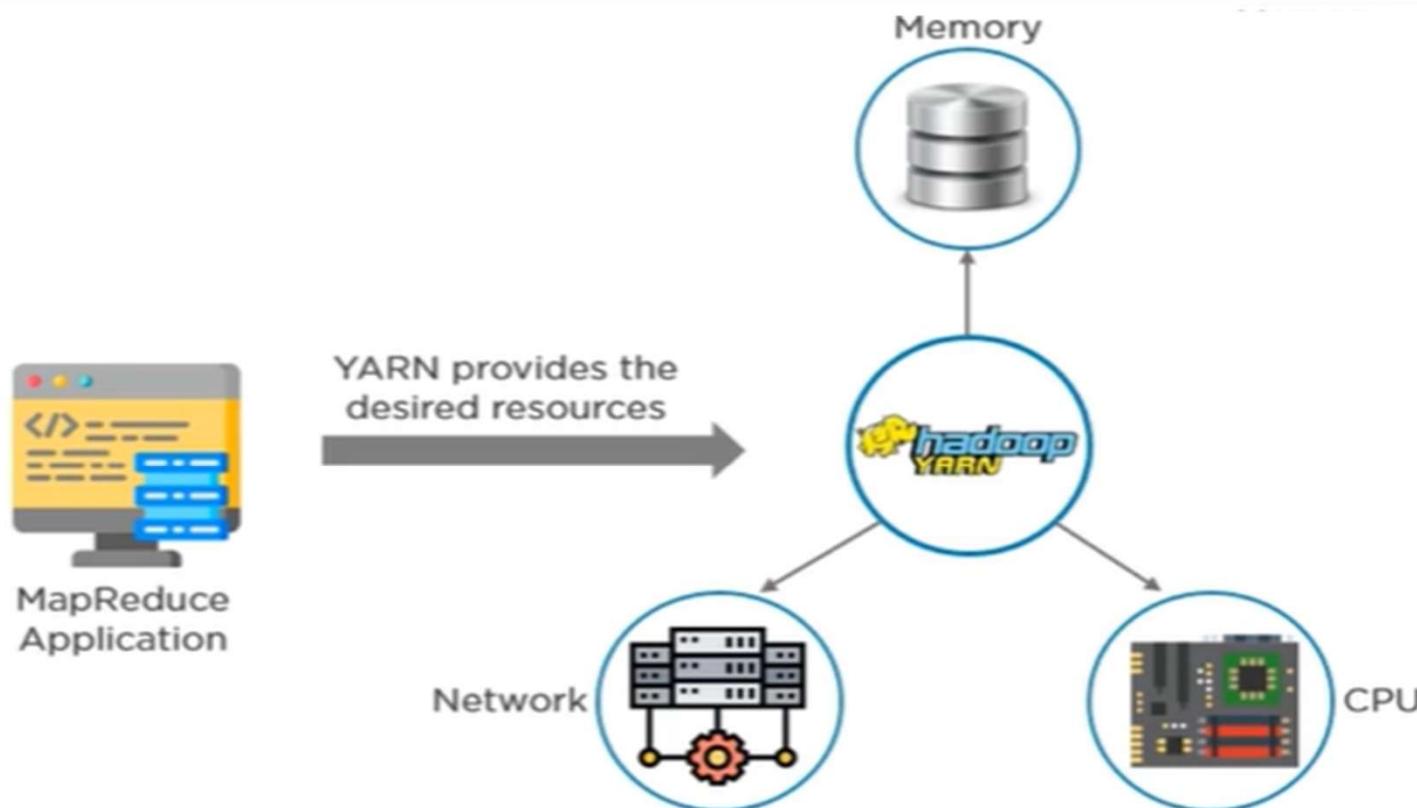


Can use open-source and proprietary data access engines and perform real-time analysis and running ad-hoc query

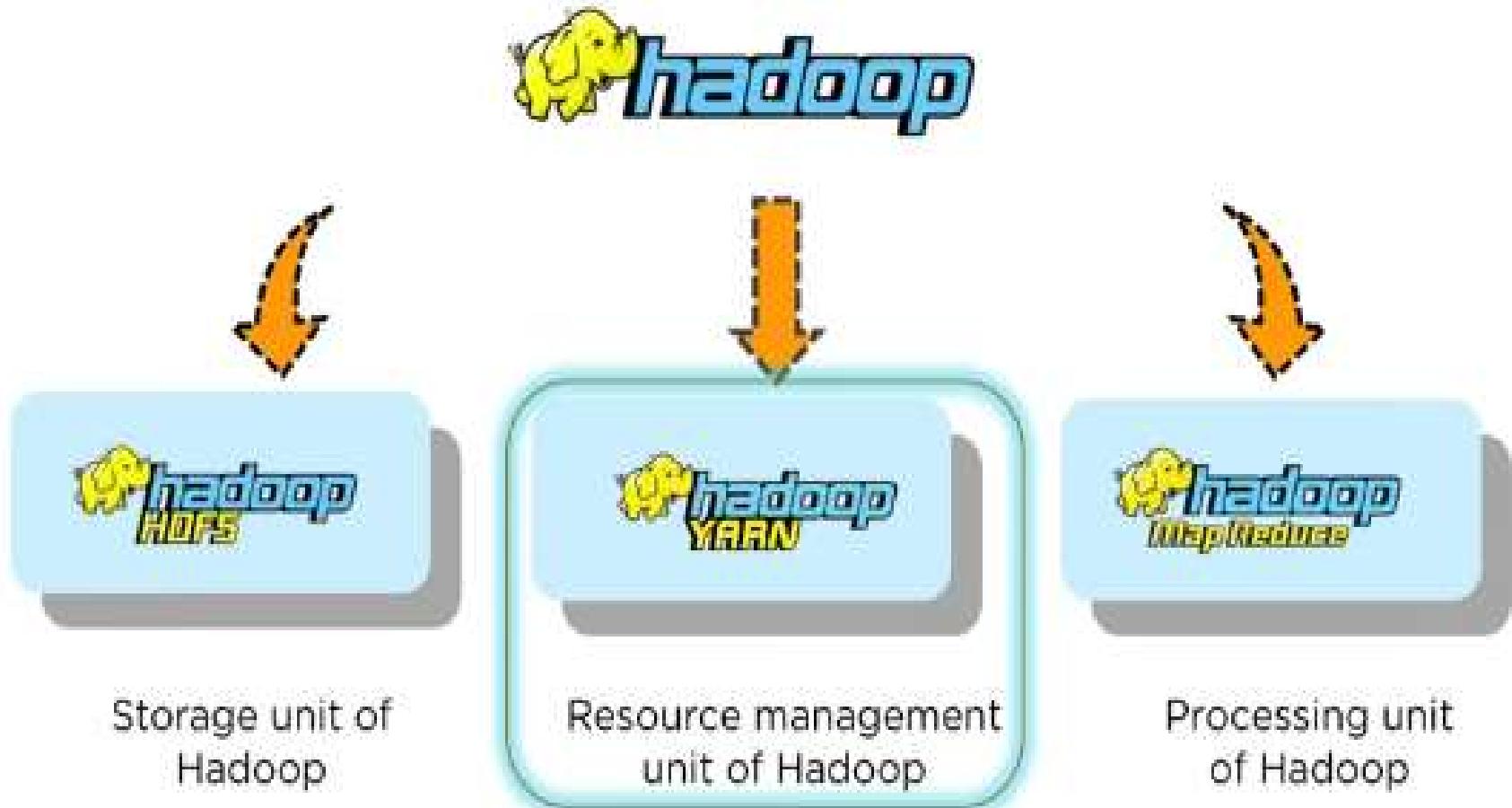
# What is YARN ?

YARN – Yet Another Resource Negotiator

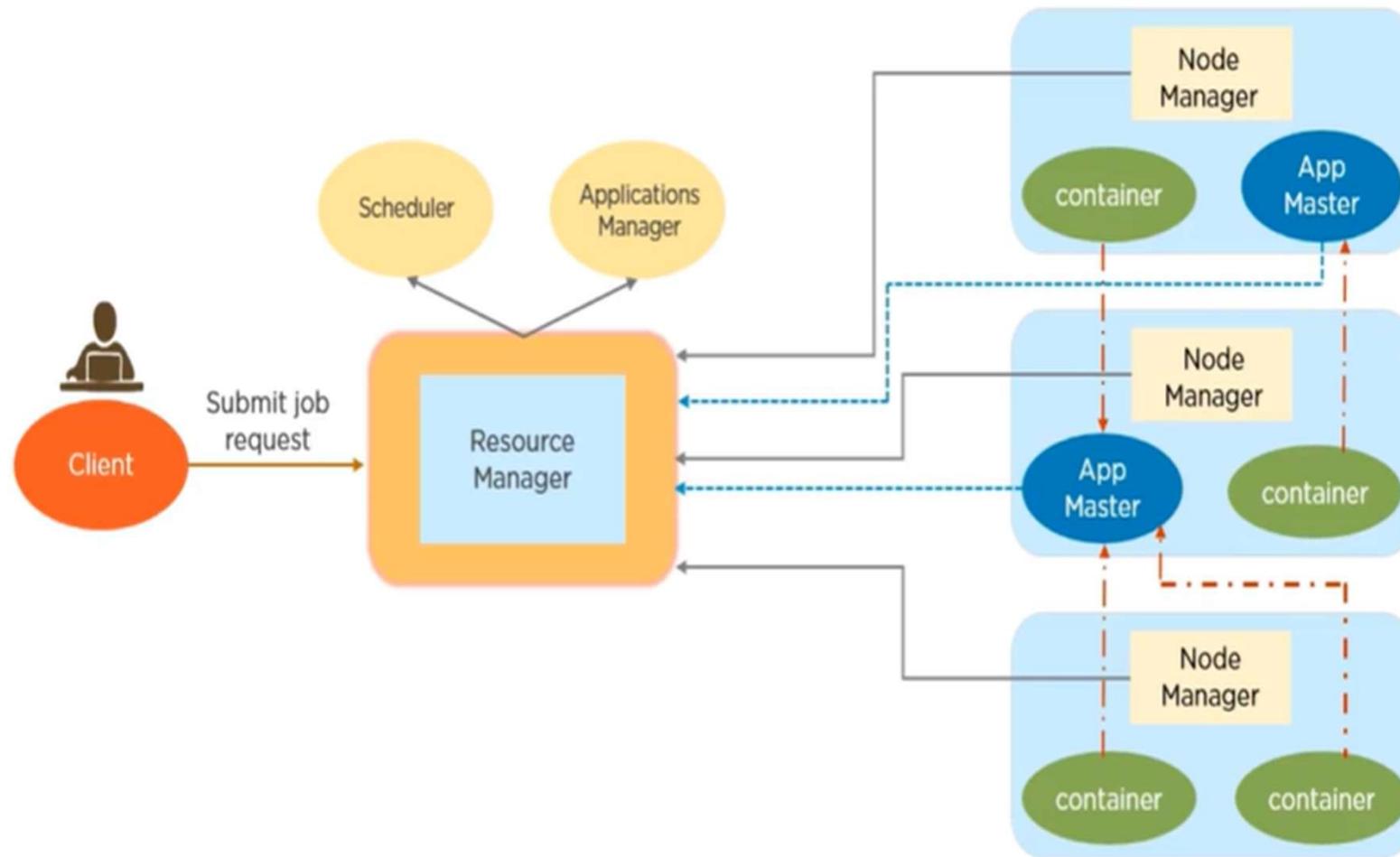
YARN is the cluster resource management layer of the Apache Hadoop Ecosystem, which schedules jobs and assigns resources



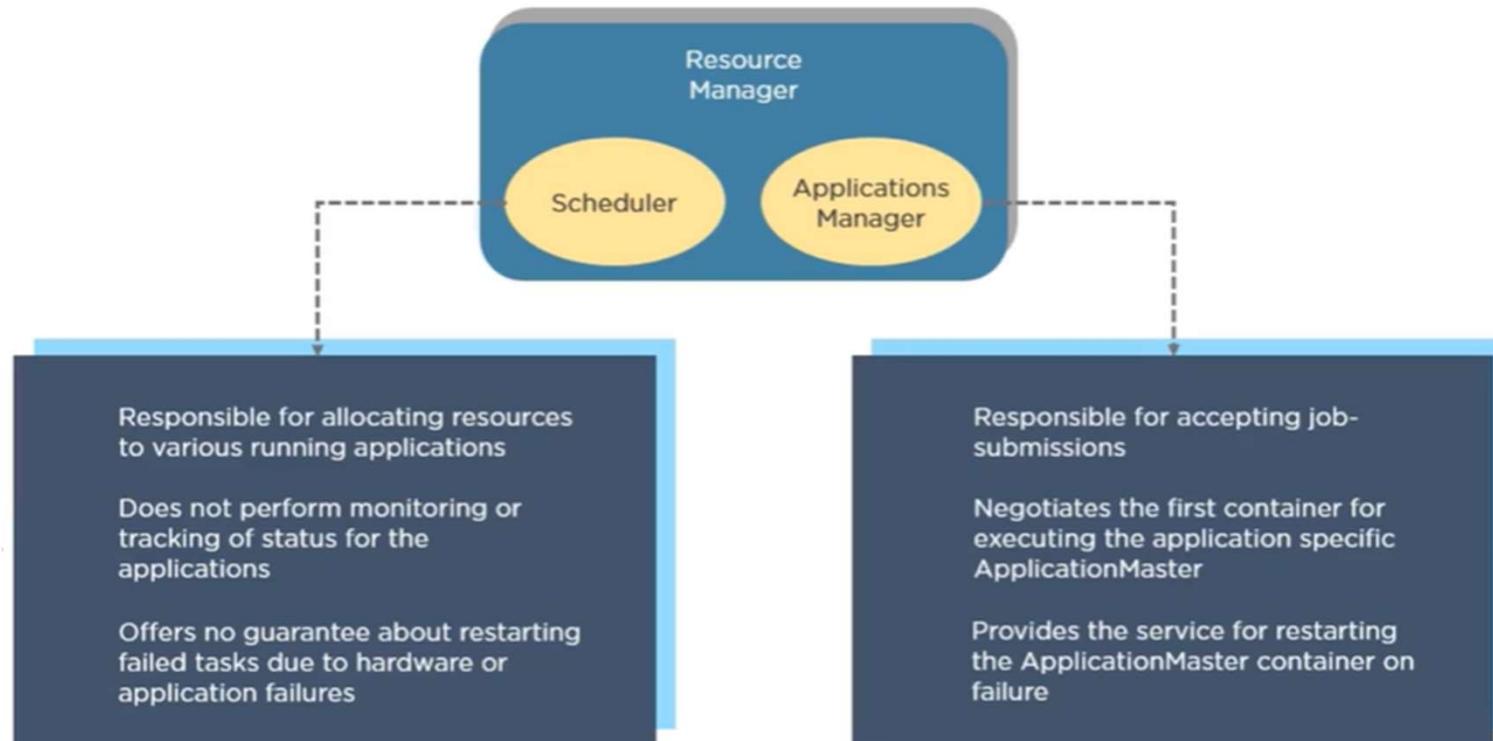
# Basic Hadoop-2 Architecture



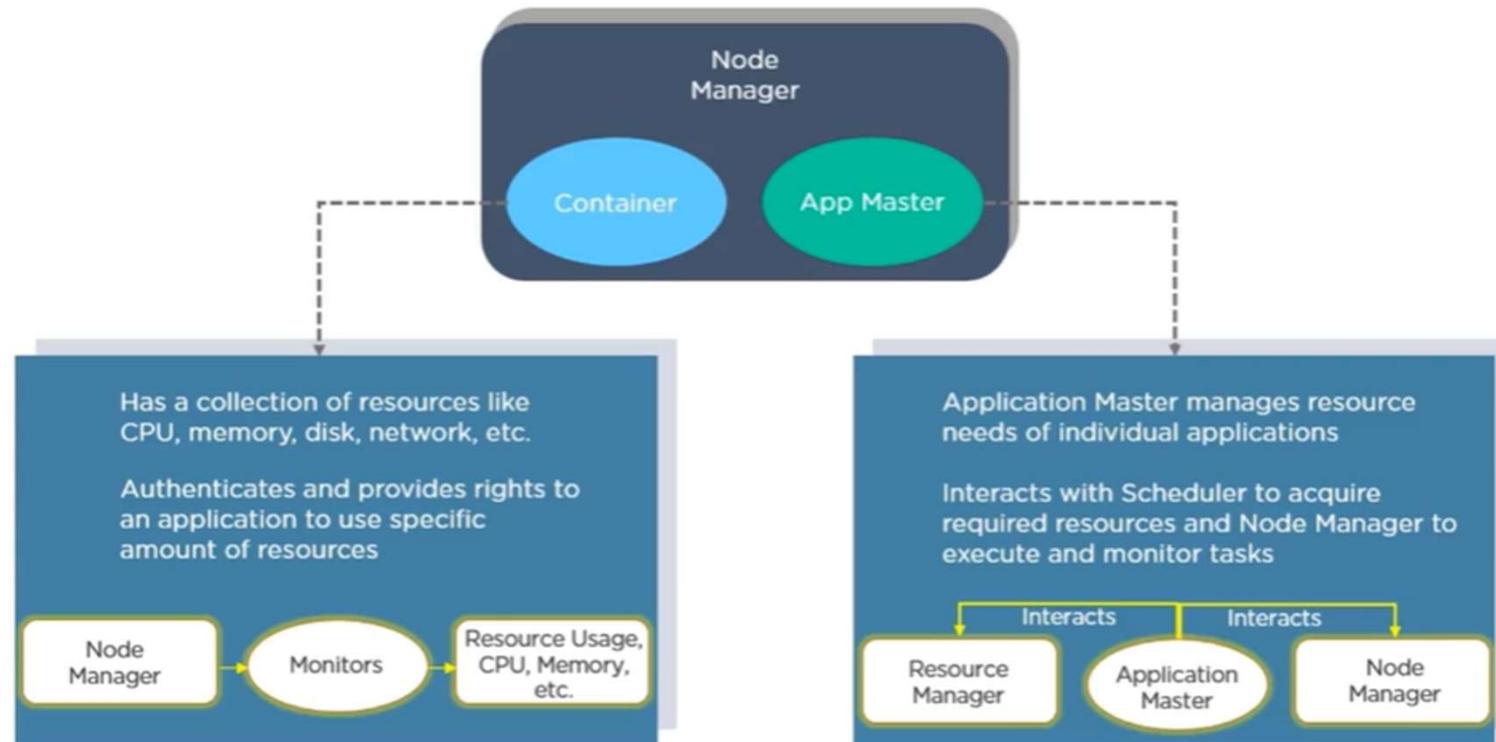
# Architecture



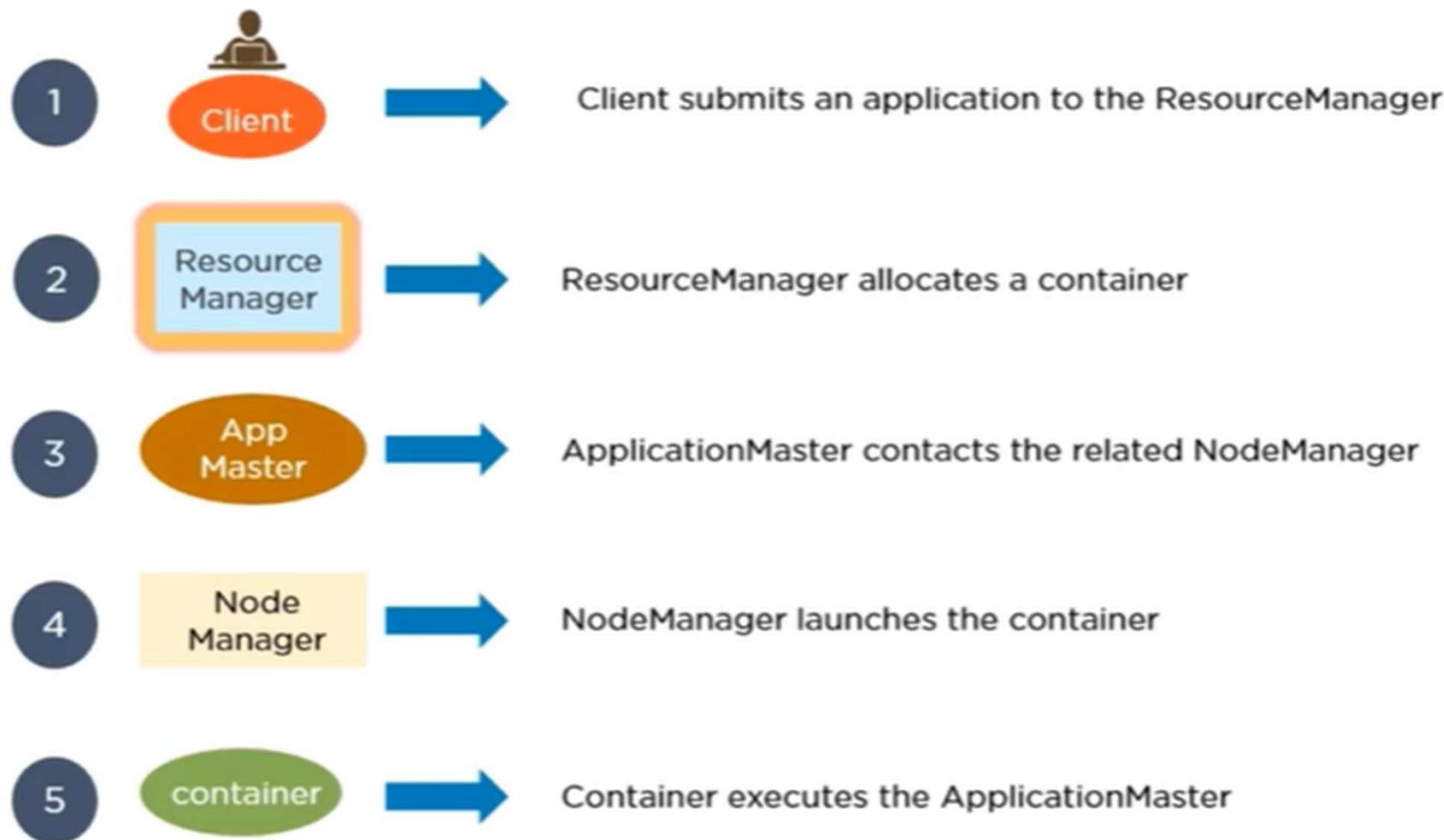
# Resource Manager



# Node Manager

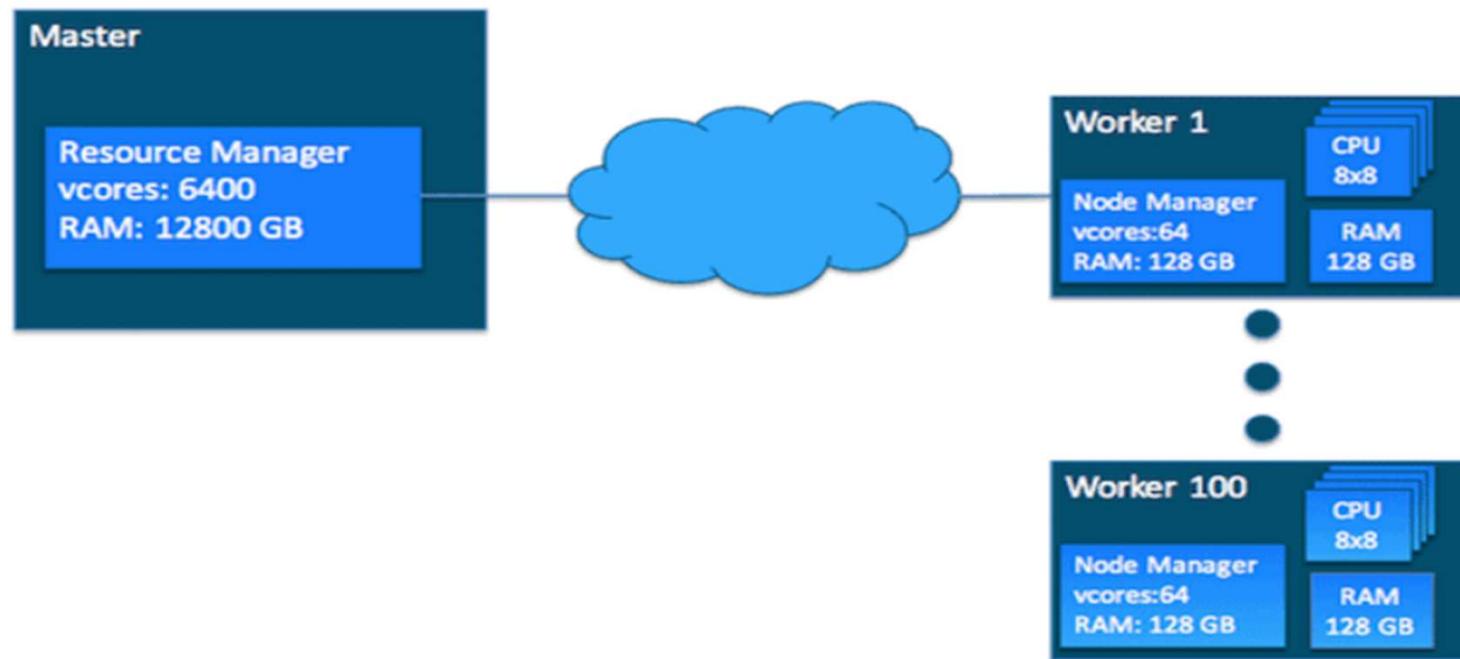


# Team work



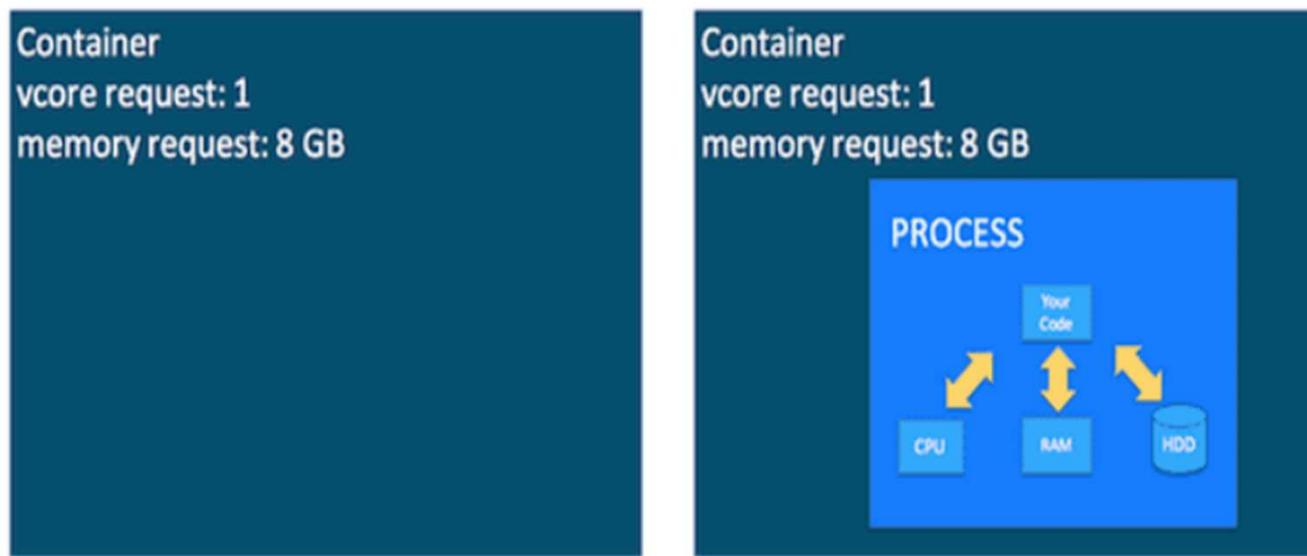
# Global View

YARN currently defines two resources, *vcores* and *memory*. Each NodeManager tracks its own local resources and communicates its resource configuration to the ResourceManager, which keeps a running total of the cluster's available resources. By keeping track of the total, the ResourceManager knows how to allocate resources as they are requested. (Vcore has a special meaning in YARN. You can think of it simply as a "usage share of a CPU core." If you expect your tasks to be less CPU-intensive (sometimes called I/O-intensive), you can set the ratio of vcores to physical cores higher than 1 to maximize your use of hardware resources.)



# Container

Containers are an important YARN concept. You can think of a container as a request to hold resources on the YARN cluster. Currently, a container hold request consists of vcore and memory, shown in Figure



Container as a hold (left), and container as a running process (right)

Once a hold has been granted on a host, the NodeManager launches a process called a *task*. The right side of Figure shows the task running as a process inside a container.

# Running Process/App Master

1. The application starts and talks to the ResourceManager for the cluster:

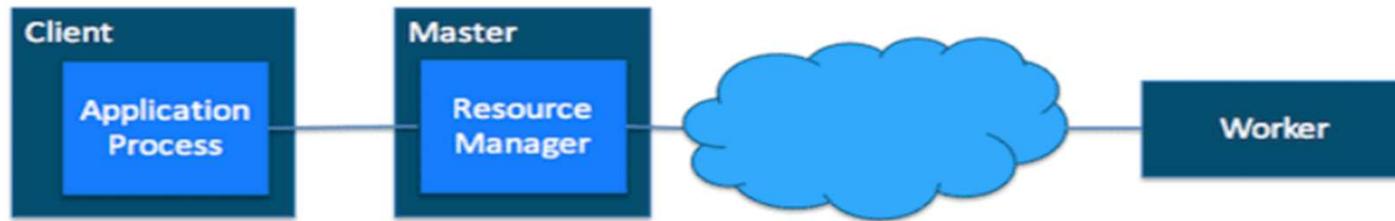


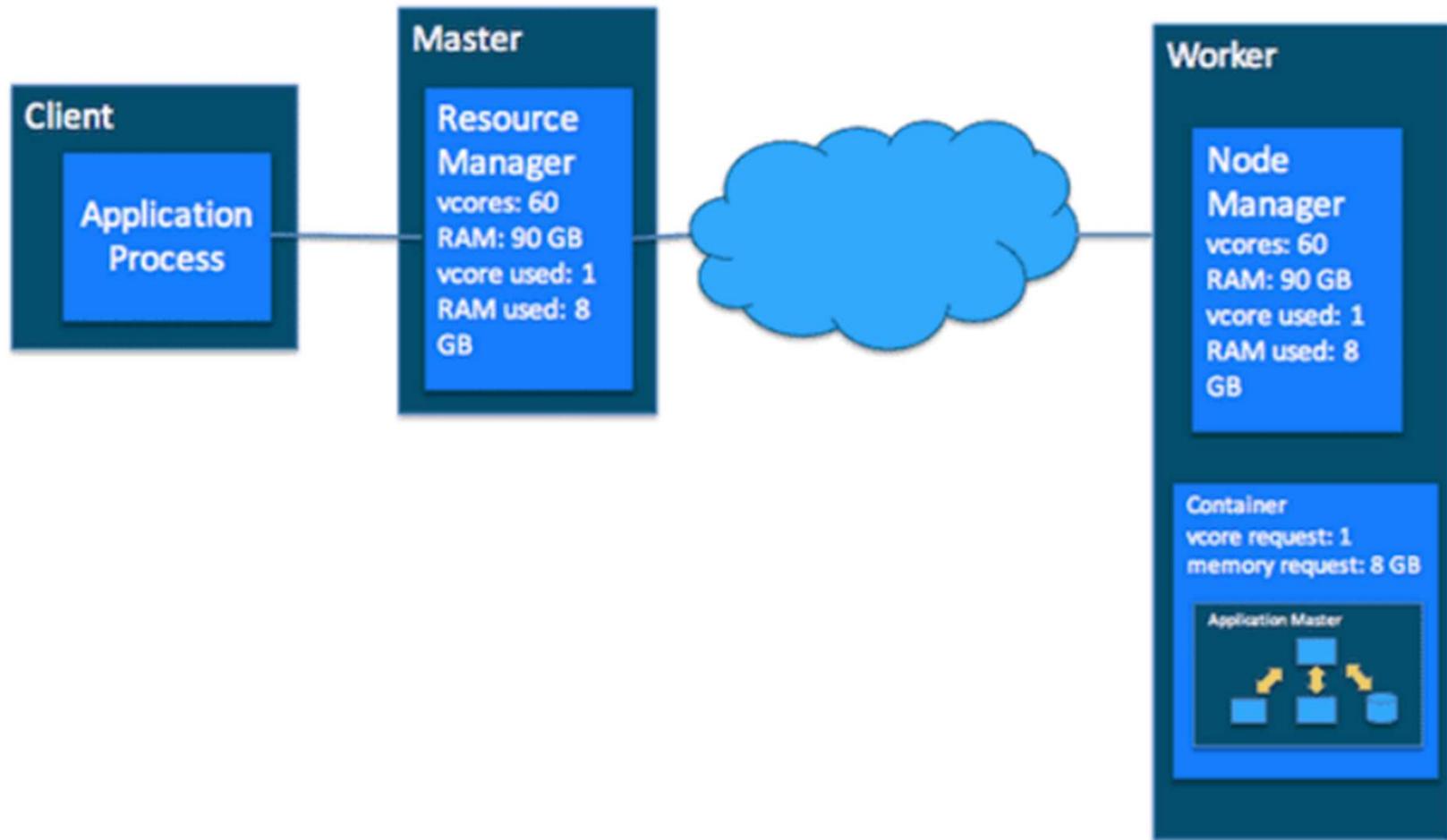
Figure 5: Application starting up before tasks are assigned to the cluster

2. The ResourceManager makes a single container request on behalf of the application:

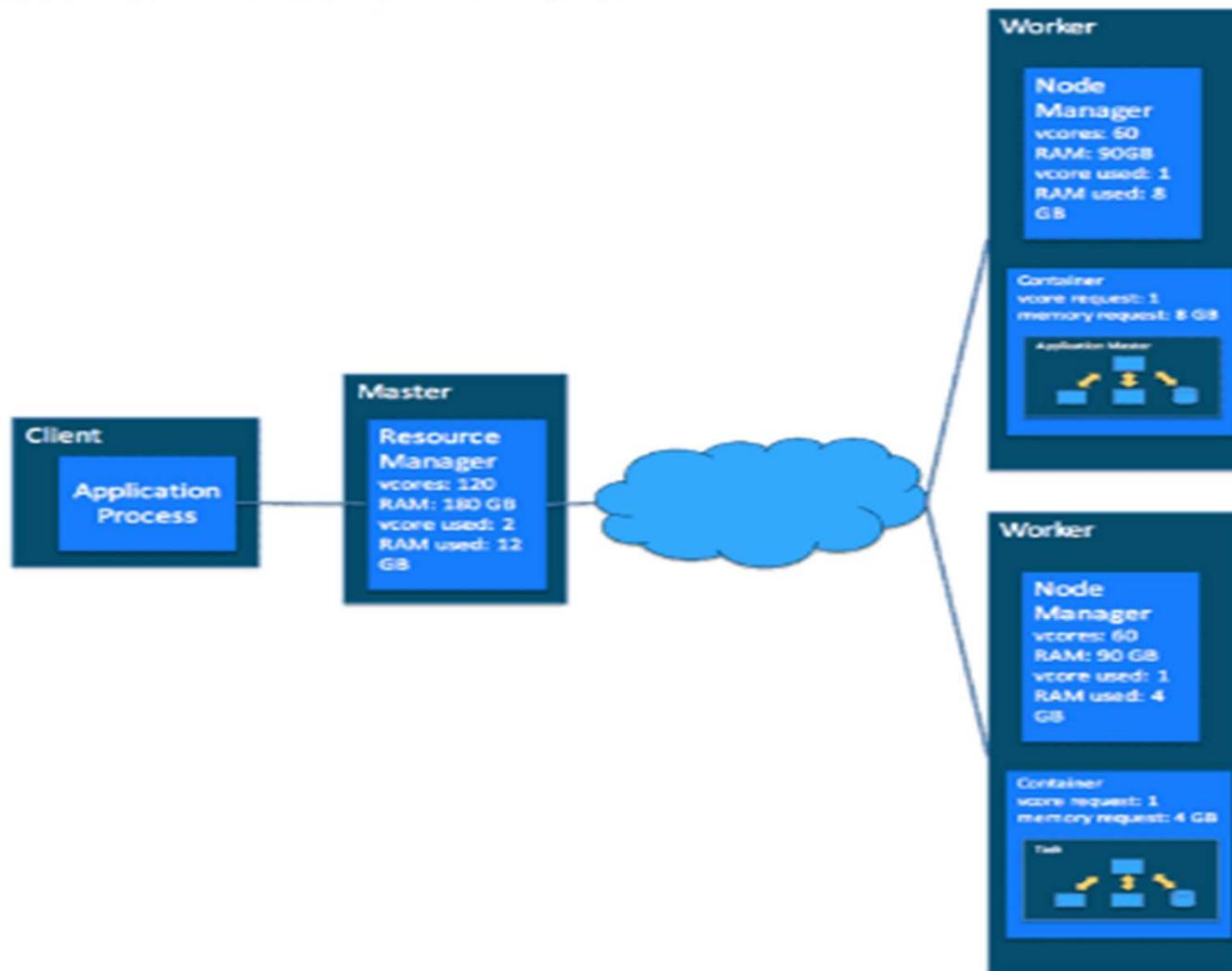


# Step 3

3. The ApplicationMaster starts running within that container:



4. The ApplicationMaster requests subsequent containers from the ResourceManager that are allocated to run tasks for the application. Those tasks do most of the status communication with the ApplicationMaster allocated in Step 3):



# Putting It Together

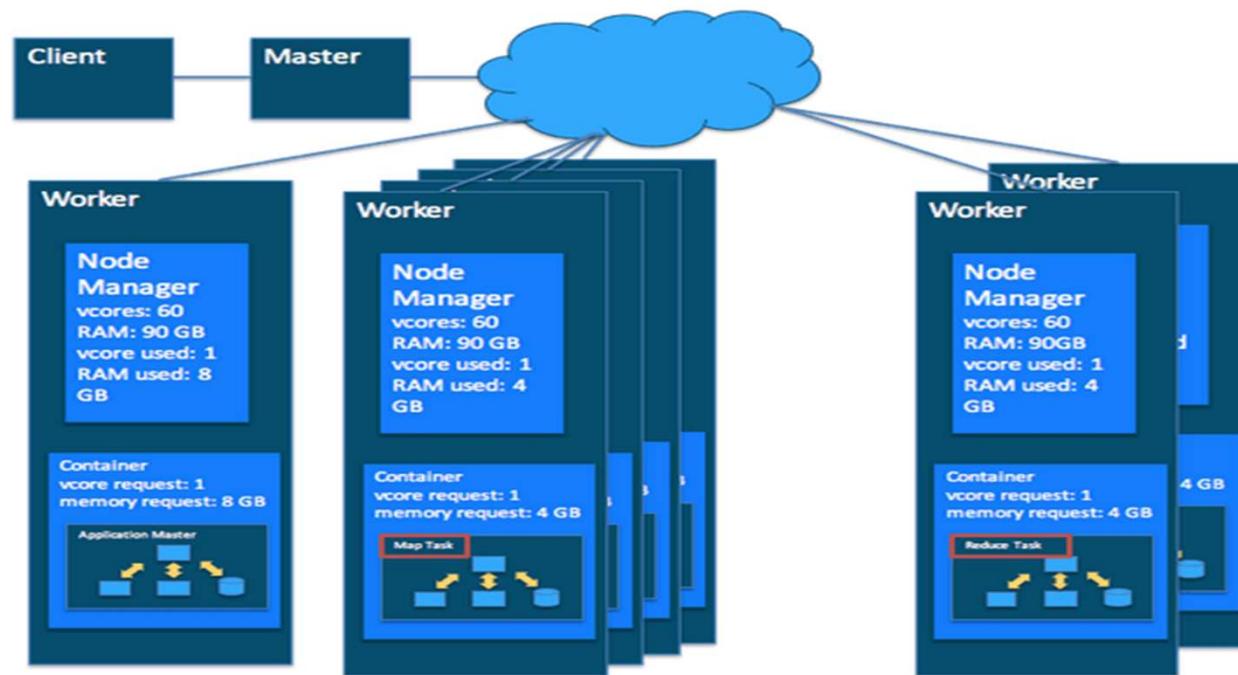


Figure 10: Merged MapReduce/YARN Application Running on a Cluster

In a MapReduce application, there are multiple map tasks, each running in a container on a worker host somewhere in the cluster. Similarly, there are multiple reduce tasks, also each running in a container on a worker host.

Simultaneously on the YARN side, the ResourceManager, NodeManager, and ApplicationMaster work together to manage the cluster's resources and ensure that the tasks, as well as the corresponding application, finish cleanly.

# YARN Installation facts

---

In reality, there are two reasons why the full set of resources on a node cannot be allocated to YARN:

1. Non-Apache Hadoop services are also required to be running on a node (overhead).
2. Other Hadoop-related components require dedicated resources and cannot be shared with YARN (such as when running CDH).

The Intel-backed group develops CDH, a distribution of Hadoop that includes several other open-source projects, such as Impala and Search. It also offers security and integration features. The Impala framework is an interactive SQL query engine that allows direct queries of data stored in HDFS, Apache HBase, or AWS S3.

# YARN in RM UI

As mentioned before, the ResourceManager has a snapshot of the available resources on the YARN cluster.

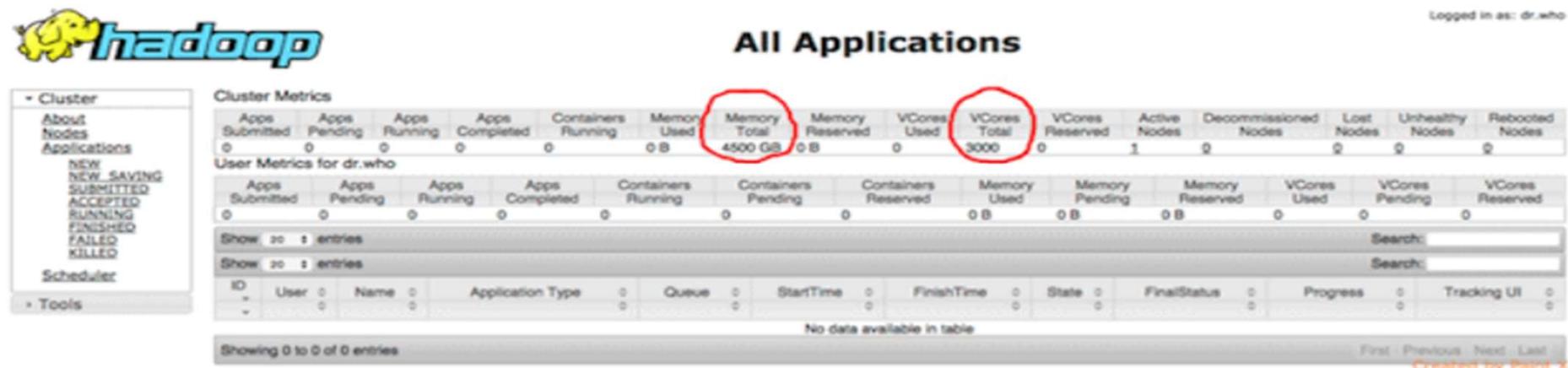
Example: Assume you have the following configuration on your 50 Worker nodes:

1. `yarn.nodemanager.resource.memory-mb = 90000`
2. `yarn.nodemanager.resource.vcores = 60`

Doing the math, your cluster totals should be:

1. memory:  $50 * 90\text{GB} = 4500\text{GB} = 4.5\text{TB}$
2. vcores:  $50 * 60 \text{ vcores} = 3000 \text{ vcores}$

On the ResourceManager Web UI page, the cluster metrics table shows the total memory and total vcores for the cluster, as seen in Figure 2 below.



The screenshot shows the "All Applications" page of the Hadoop ResourceManager Web UI. The top navigation bar includes the Hadoop logo and the text "Logged in as: dr.who". On the left, there's a sidebar with sections for Cluster (About, Nodes, Applications, Scheduler), Applications (NEW, NEW\_SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, KILLED), and Tools. The main content area is titled "All Applications" and contains two tables under "Cluster Metrics". The first table shows overall cluster statistics: Apps Submitted (0), Apps Pending (0), Apps Running (0), Apps Completed (0), Containers Running (0), Memory Used (0 B), Memory Total (4500 GB), Memory Reserved (0 B), Vcores Used (0), Vcores Total (3000), Vcores Reserved (0), Active Nodes (1), Decommissioned Nodes (0), Lost Nodes (0), Unhealthy Nodes (0), and Rebooted Nodes (0). The second table provides user-specific metrics for "dr.who": Apps Submitted (0), Apps Pending (0), Apps Running (0), Apps Completed (0), Containers Running (0), Containers Pending (0), Containers Reserved (0), Memory Used (0 B), Memory Pending (0 B), Memory Reserved (0 B), Vcores Used (0), Vcores Pending (0), and Vcores Reserved (0). Below these tables, there are search fields and links for "Show 20 entries" and "Search". A note at the bottom states "No data available in table". At the very bottom, it says "Showing 0 to 0 of 0 entries" and "First Previous Next Last". The footer indicates the page was "Created by Pinal X".

# Container Config

At this point, the YARN Cluster is properly set up in terms of Resources. YARN uses these resource limits for allocation, and enforces those limits on the cluster.

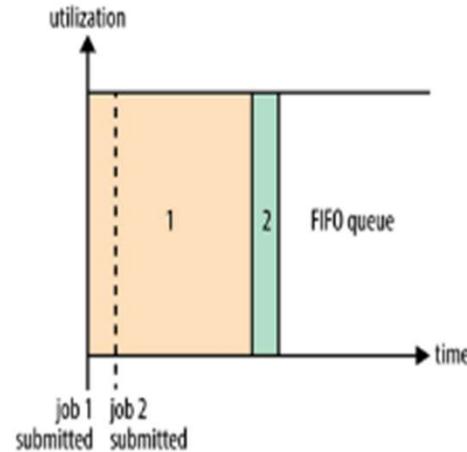
- YARN Container Memory Sizing
  - Minimum: `yarn.scheduler.minimum-allocation-mb`
  - Maximum: `yarn.scheduler.maximum-allocation-mb`
- YARN Container VCore Sizing
  - Minimum: `yarn.scheduler.minimum-allocation-vcores`
  - Maximum: `yarn.scheduler.maximum-allocation-vcores`
- YARN Container Allocation Size Increments
  - Memory Increment: `yarn.scheduler.increment-allocation-mb`
  - VCore Increment: `yarn.scheduler.increment-allocation-vcores`

# Restrictions

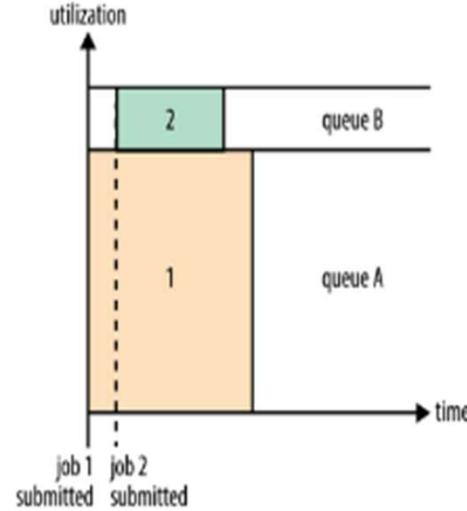
- Memory properties:
  - Minimum required value of 0 for `yarn.scheduler.minimum-allocation-mb`.
  - Any of the memory sizing properties must be less than or equal to `yarn.nodemanager.resource.memory-mb`.
  - Maximum value must be greater than or equal to the minimum value.
- VCore properties:
  - Minimum required value of 0 for `yarn.scheduler.minimum-allocation-vcores`.
  - Any of the vcore sizing properties must be less than or equal to `yarn.nodemanager.resource.vcores`.
  - Maximum value must be greater than or equal to the minimum value.
  - Recommended value of 1 for `yarn.scheduler.increment-allocation-vcores`. Higher values will likely be wasteful.

# Scheduling YARN

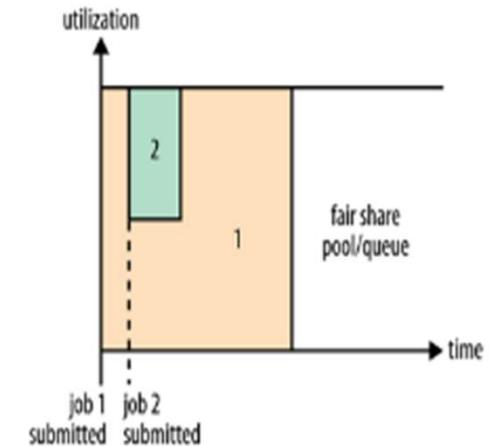
i. FIFO Scheduler



ii. Capacity Scheduler



iii. Fair Scheduler



Jump To Next PPT

# Add YARN Service

**Home**

Status All Health Issues Configuration  All Recent Commands

**Cluster 1** (CDH 5.12.1, Parcels)

-  Hosts
-  HDFS
-  CloudBees

**Actions**

-  Add Service
- Start
- Stop
- Restart
- Rolling Restart
- Deploy Client Configuration

**Charts**

**Cluster CPU**



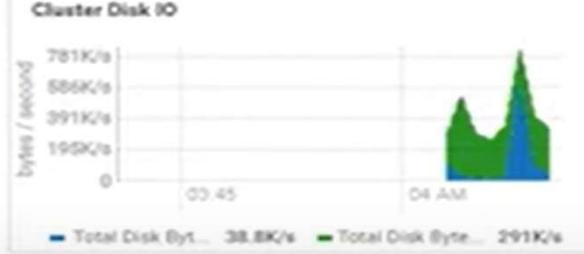
Percent

0%, 50%, 100%

03:45 04 AM

Cluster 1, Host CPU Usage Across Hosts 5.1%

**Cluster Disk IO**



bytes / second

0, 199K/s, 391K/s, 585K/s, 781K/s

03:45 04 AM

Total Disk Byte... 38.8K/s Total Disk Byte... 291K/s

without directly providing AWS credentials, subject to having the proper permissions defined via Sentry protocol.

 Sentry	Sentry service stores authorization policy metadata and provides clients concurrent and secure access to data.
 Solr	Solr is a distributed service for indexing and searching data stored in HDFS.
 Spark	Apache Spark is an open source cluster computing system. This service runs Spark as an application on the cluster.
 Spark (Standalone)	Apache Spark is an open source cluster computing system. This is the standalone version of the service management. Cloudera recommends using Spark on YARN instead of this standalone version.
 Sqoop 1 Client	Configuration and connector management for Sqoop 1.
 Sqoop 2	Sqoop is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores. The version supported by Cloudera Manager is Sqoop 2.
 YARN (MR2 Included)	Apache Hadoop MapReduce 2.0 (MRv2), or YARN, is a data computation framework that supports MapReduce and streaming processing. It includes support for HDFS and other storage systems.
 ZooKeeper	Apache ZooKeeper is a centralized service for maintaining and synchronizing configuration data.

## Add YARN (MR2 Included) Service to Cluster 1

### Customize Role Assignments for YARN (MR2 Included)

You can customize the role assignments for your new service here, but note that if assignments are made incorrectly, such as assigning too many roles to a host, it may suffer.

You can also view the role assignments by host.

[View By Host](#)

**RM** ResourceManager × 1 New

ip-172-31-9-240.ap-southeast-1.compute.internal

**JHS** JobHistory Server × 1 New

ip-172-31-0-223.ap-southeast-1.compute.internal

**NM** NodeManager × 3 New

ip-172-31-0-223.ap-southeast-1.compute.internal

Select hosts for a new or existing role. The host list is filtered to remove hosts that are not valid candidates; these include hosts that are unhealthy, members of other clusters, or have incompatible version of CDH installed on them.

Enter hostnames: host01, host[01-10], IP addresses or rack.

[Search](#)

Hostname	IP Address	Rack	Cores	Physical Memory	Existing Roles	Added Role
<input type="checkbox"/> ip-172-31-0-223.ap-southeast-1.compute.internal	172.31.0.223	/default	2	7.7 GiB	B DN AP SM	NM
<input type="checkbox"/> ip-172-31-14-33.ap-southeast-1.compute.internal	172.31.14.33	/default	2	7.7 GiB	DN HM	NM
<input type="checkbox"/> ip-172-31-5-165.ap-southeast-1.compute.internal	172.31.5.165	/default	2	7.7 GiB	DN SNN RM	NM
<input checked="" type="checkbox"/> ip-172-31-9-240.ap-southeast-1.compute.internal	172.31.9.240	/default	2	7.7 GiB	NN ES	RM JHS

# Default config

## Add YARN (MR2 Included) Service to Cluster 1

Review Changes

### Default Configuration

#### NodeManager Local Directories

yarn.nodemanager.local-dirs

#### NodeManager Default Group

/yarn/nm

#### Enable Container Usage Metrics Collection

YARN (MR2 Included) (Service-Wide)

#### Container Usage MapReduce Job User

YARN (MR2 Included) (Service-Wide)

#### Cloudera Manager Container Usage Metrics Directory

YARN (MR2 Included) (Service-Wide)

/tmp/cmYarnContainerMetrics

#### Container Usage Output Directory

YARN (MR2 Included) (Service-Wide)

/tmp/cmYarnContainerMetricsAggregate

# Adding Services

## Add YARN (MR2 Included) Service to Cluster 1

### First Run Command

Status  Running  Oct 29, 4:13:06 AM

**Adding the service...it may take some time**

**Completed 3 of 4 step(s).**

Show All Steps  Show Only Failed Steps  Show Only Running Steps

> <input checked="" type="checkbox"/> Ensuring that the expected software releases are installed on hosts.	Successfully completed 1 steps.
> <input checked="" type="checkbox"/> Deploying Client Configuration	 Cluster 1 
> <input checked="" type="checkbox"/> Creating DFS directories required for YARN	Successfully deployed all client configurations.
> <input type="checkbox"/> Start YARN (MR2 Included)	 YARN (MR2 Included) 

# Contd..

```
[root@ip-172-31-0-223 ~]# service ntpd start
Starting ntpd: [ OK ]
[root@ip-172-31-0-223 ~]# service ntpd status
ntpd (pid 5061) is running...
[root@ip-172-31-0-223 ~]# hdfs dfs -ls /user/
ls: `/user/': No such file or directory
You have new mail in /var/spool/mail/root
[root@ip-172-31-0-223 ~]# hdfs dfs -ls /
Found 1 items
drwxrwxrwt - hdfs supergroup 0 2017-10-29 04:02 /tmp
[root@ip-172-31-0-223 ~]# hdfs dfs -ls /
Found 2 items
drwxrwxrwt - hdfs supergroup 0 2017-10-29 04:13 /tmp
drwxr-xr-x - hdfs supergroup 0 2017-10-29 04:13 /user
```

Directory created by YARN

# Service added Successfully

**Cluster 1 (CDH 5.12.1, Parcels)**

- Hosts: 3
- HDFS: 1
- YARN (MR2 Included)** (YARN SERVICE ADDED)

**Cloudera Management Service**

- Cloudera M...: 2

### Charts

30m 1h 2h 6h

**Cluster CPU**



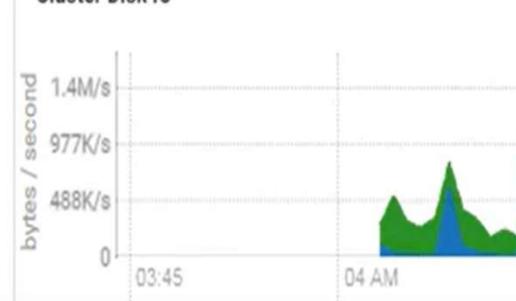
percent

0%, 50%, 100%

03:45 04 AM

Cluster 1, Host CPU Usage Across Hosts 9.9%

**Cluster Disk IO**



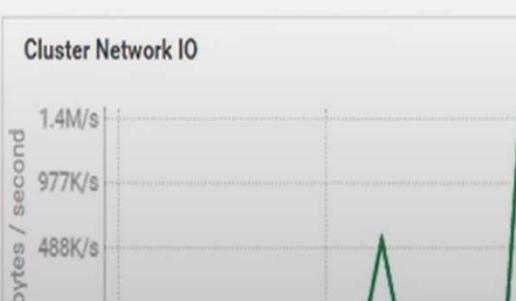
bytes / second

0, 488K/s, 977K/s, 1.4M/s

03:45 04 AM

Total Disk Byte... 1.3M/s Total Disk Byte... 471K/s

**Cluster Network IO**



bytes / second

0, 488K/s, 977K/s, 1.4M/s

**HDFS IO**



bytes / second

0, 2b/s, 4b/s

**Home** Status All Health Issues 04 Configuration 5 All Recent Commands

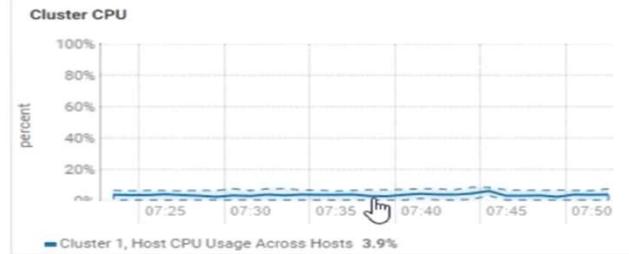
You are running Cloudera Manager in non-production mode, which uses an embedded PostgreSQL database. Switch to using a supported external database before moving into production. [More Details](#)

**Cluster 1** (CDH 5.14.0, Parcels)

- 9 Hosts 5
- Flume
- HBase
- HDFS
- Hive
- Hue
- Impala
- Kafka
- Key-Value St...
- Impala
- Kafka
- Key-Value St...
- Oozie
- Solr
- Spark
- Spark 2
- Spoon 2
- YARN (MR2 I...)**
- ZooKeeper

### Charts

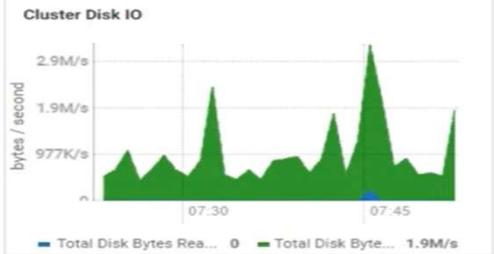
**Cluster CPU**



percent

Cluster 1, Host CPU Usage Across Hosts 3.9%

**Cluster Disk IO**



bytes / second

Total Disk Bytes Read... 0 Total Disk Byte... 1.9M/s

**Cluster Network IO**



bytes / second

Cluster 1, Host CPU Usage Across Hosts 3.9%

**HDFS IO**



bytes / second

Total Disk Bytes Read... 0 Total Disk Byte... 1.9M/s

**Cluster Network IO**



bytes / second

Total Bytes Rec... 1.2M/s Total Bytes Tra... 1.2M/s

**HDFS IO**



bytes / second

Total Bytes Read Acr... 0 Total Bytes Writ... 1.9b/s

**Completed Impala Queries**

ind

**physical memory**

55.9G

Feedback

# Status page

**Status** instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI ▾ Quick Links ▾

**Health Tests** [Create Trigger](#)

 Show 3 **good**

 YARN Container Usage Aggregation [Suppress...](#)

This health test is disabled because container usage metric collection is disabled for YARN.

**Status Summary**

JobHistory Server	 1 Good Health
NodeManager	 3 Good Health
ResourceManager	 1 Good Health
Hosts	 4 Good Health

**Charts**

**Applications Running (Cumulative)**



applications

03:45 04 AM

root (YARN (MR2 Included)), Applications Running (... 0

**Total Containers Running Across NodeManagers ...**

containers

NO DATA

# Instances/Roles

 YARN (MR2 Included) (Cluster 1) Actions ▾ Apr 14, 7:53 PM UTC

Status Instances Configuration Commands Applications Resource Pools Charts Library Web UI ▾ Quick Links ▾

Search 

Role Type	State	Host	Commission State	Role Group
JobHistory Server	Started	ip-10-0-1-20.ec2.internal	Commissioned	JobHistory Server Default Group
NodeManager	Started	ip-10-0-1-10.ec2.internal	Commissioned	NodeManager Default Group
NodeManager	Started	ip-10-0-2-12.ec2.internal	Commissioned	NodeManager Group 1
NodeManager	Stopped	ip-10-0-2-15.ec2.internal	Decommissioned	NodeManager Default Group
NodeManager	Started	ip-10-0-2-11.ec2.internal	Commissioned	NodeManager Group 1
NodeManager	Stopped	ip-10-0-2-14.ec2.internal	Decommissioned	NodeManager Group 2
NodeManager	Started	ip-10-0-2-13.ec2.internal	Commissioned	NodeManager Default Group
ResourceManager (Active)	Started	ip-10-0-1-20.ec2.internal	Commissioned	ResourceManager Default Group

**Filters**

- STATUS**
  - Stopped 2
  - Good Health 6
- COMMISSION STATE**
- MAINTENANCE MODE**
- RACK**
- ROLE GROUP**
- ROLE TYPE**
- STATE**



# Configuration

 YARN (MR2 Included) (Cluster 1) Actions ▾ Apr 14, 7:53 PM UTC

Status Instances Configuration Commands Applications Resource Pools Charts Library Web UI ▾ Quick Links ▾

Search I Switch to the classic layout Role Groups

**Filters** Read-only: Requires additional authorization Show All Descriptions

**SCOPE**

Service	Scope	Description
HDFS Service	YARN (MR2 Included) (Service-Wide)	hdfs
ZooKeeper Service	YARN (MR2 Included) (Service-Wide)	zookeeper

**CATEGORY**

Category	Setting	Description
Advanced	yarn.acl.enable	Enable ResourceManager ACLs
Compression		
Logs		
Main		
Monitoring		

Admin API VADM / YARN (MR2 Included) / Overview Metrics

# Resource Manager UI



Logged in as: dr

## All Applications

Cluster	
<a href="#">About</a>	
<a href="#">Nodes</a>	
<a href="#">Applications</a>	
<a href="#">NEW</a>	
<a href="#">NEW SAVING</a>	
<a href="#">SUBMITTED</a>	
<a href="#">ACCEPTED</a>	
<a href="#">RUNNING</a>	
<a href="#">FINISHED</a>	
<a href="#">FAILED</a>	
<a href="#">KILLED</a>	
<a href="#">Scheduler</a>	
<a href="#">Tools</a>	

### Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved
0	0	0	0	0	0 B	3 GB	0 B	0	6	0

### Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
3	0	0	0	0	0

### User Metrics for dr.who

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	Vcores Used	Vcores Pending	Vcore Reserved
0	0	0	0	0	0	0	0 B	0 B	0 B	0	0	0

Show 20 ▾ entries

Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	Allocated Memory MB	Progress	Tracking UI

# Job History UI



Logged in as

## JobHistory

Retired Jobs

Show	20	entries	Search:								
Submit Time	Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Completed	Reduces Total	Redu

No data available in table

Submit Time	Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Comple	Reduces Tota	Redu
-------------	------------	-------------	--------	------	------	-------	-------	------------	-------------	--------------	------

Showing 0 to 0 of 0 entries

First Previous Next

# YARN Tuning



Why Cloudera   Products   Services & Support



This is the documentation for Cloudera Enterprise 5.4.x. Documentation for other versions is available at [Cloudera Docs](#)

Documentation > [Cloudera Installation and Upgrade](#) > [Installing Cloudera Manager and CDH](#) > [Installing and Deploying CDH Using the Command Line](#)

[View All Categories](#)

- ▶ Cloudera Introduction
  - Cloudera Release Notes
- ▶ Cloudera QuickStart
- ▼ Cloudera Installation and Upgrade
  - ▶ Installation Requirements for Cloudera Manager, Cloudera Navigator, and CDH 5
- ▼ Installing Cloudera Manager and CDH
  - Java Development Kit Installation
  - ▶ Installing Cloudera Manager, CDH,

## Tuning YARN

This topic applies to YARN clusters only, and describes how to tune and optimize YARN for yo

**Note:** Download the Cloudera [YARN tuning spreadsheet](#) to help calculate YARN configura  
short video overview, see [Tuning YARN Applications](#).

## Overview

This overview provides an abstract description of a YARN cluster and the goals of YARN tunin

# Machine Configuration

## STEP 1: Worker Host Configuration

Enter your likely machine configuration in the input boxes below. If you are uncertain what machines you plan on buying, put in some minimum values that will suit what you expect to buy.

Host Components	Quantity	Description
RAM	8	Gigabytes
CPU	2	3 CPUs: 6 cores, 3.5 GHz, 15MB cache
HDD (Hard Disk Drive)	36	.2x3TB SATA III Hard Drives in JBOD Configuration
Ethernet	2	. Gigabit Ethernet

## STEP 2: Worker Host Planning



to allocate resources, mainly CPU and memory, to the various software components that run on the host.

	Service	Category	CPU (cores)	Memory (M)	Notes
16	Operating System	Overhead	1	1024	Most operating systems use 4-8GB minimum.
17	Task overhead	Overhead	0	0	Allow additional memory overhead for task buffers such as
18	Cloudera Manager agent	Overhead	0	1024	Allocate 1GB for Cloudera Manager agents, which track res
19	Other services	Overhead	0	0	Enter the required cores or memory for services not listed a
20	HDFS DataNode	CDH	0	1024	Allocate 1GB for the HDFS DataNode.
21	Impala daemon	CDH	0	0	(Optional Service) Suggestion: Allocate at least 16GB mem
22	Hbase RegionServer	CDH	0	0	(Optional Service) Suggestion: Allocate no more than 12-16
23	Solr Server	CDH	0	0	(Optional Service) Suggestion: Minimum 1GB for Solr server
24	YARN NodeManager	CDH	0	1024	Allocate 1GB for the YARN NodeManager.
25	Available Resources		1	4096	
26	Physical Cores to Vcores Multiplier		4		Set this ratio based on the expected number of concurrent t
27	YARN Available Vcores		4		This value will be used in STEP 4 for YARN Configuration
28	YARN Available Memory			4096	This value will be used in STEP 4 for YARN Configuration
29					
30					
31					
32					
33					
34	STEP 3: Cluster Size				
	Cluster Configuration	YARN Configuration	MapReduce Configuration		

# Details of worker nodes

Apache RegionServer	CDH	0	0 (Optional Service) Suggestion:
HDFS Server	CDH	0	0 (Optional Service) Suggestion:
YARN NodeManager	CDH	0	1024 Allocate 1GB for the YARN Node Manager
Available Resources		1	4096
Physical Cores to Vcores Multiplier		4	Set this ratio based on the experience
YARN Available Vcores		4	This value will be used in STEP 2
YARN Available Memory		4096	This value will be used in STEP 3

## STEP 3: Cluster Size

Enter the number of nodes you have (or expect to have) in the cluster

		Quantity			
Number of Worker Hosts in the cluster		3			

# Make note of node statistics

**cloudera MANAGER** Clusters ▾ **Hosts ▾** Diagnostics ▾ Audits Charts ▾ Backup ▾ Administration ▾   Search Support ▾ admin ▾

All Hosts Configuration Add New Hosts to Cluster Re-run Upgrade Wizard Inspect All Hosts

**Filters**

Search 

**STATUS** Actions for Selected ▾ Columns: 11 Selected ▾

	IP	Roles	Commission State	Last Heartbeat	Load Average	Cores	Disk Usage	Physical Memory	Swap Space
0-	172.31.0.223	5 Role(s)	Commissioned	9.62s ago	0.06 0.11 0.10	2	9 GiB / 47.4 GiB	1.8 GiB / 7.7 GiB	0 B / 816 MiB
14-	172.31.14.33	3 Role(s)	Commissioned	9.75s ago	0.00 0.01 0.07	2	8.9 GiB / 47.4 GiB	1.5 GiB / 7.7 GiB	0 B / 816 MiB
5-	172.31.5.165	4 Role(s)	Commissioned	9.51s ago	0.21 0.07 0.09	2	9.4 GiB / 47.4 GiB	1.8 GiB / 7.7 GiB	0 B / 816 MiB
9-	172.31.9.240	4 Role(s)	Commissioned	9.18s ago	0.06 0.04 0.01	2	11 GiB / 47.4 GiB	3.9 GiB / 7.7 GiB	0 B / 816 MiB



## YARN Configuration

### STEP 4: YARN Configuration on Cluster

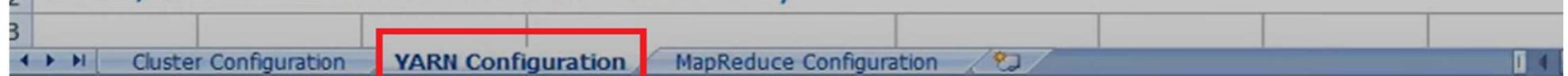
These are the first set of configuration values for your cluster. You can set these values in YARN > Configuration

**Go To These Locations and Get the Values**

YARN Configuration Property	Value
yarn.nodemanager.resource.cpu-vcores	4 Copied from STEP 2 "Av
yarn.nodemanager.resource.memory-mb	4096 Copied from STEP 2 "Av

### STEP 5: Verify YARN Settings on Cluster

Go to the Resource Manager Web UI (usually <http://<ResourceManagerIP>:8088/>) and verify that "Memory Total" and "Vcores Total" matches the values above. If your machine has no bad nodes, then the numbers should match exactly.



# Memory Available update

cloudera MANAGER Clusters ▾ Hosts ▾ Diagnostics ▾ Audits Charts ▾ Backup ▾ Administration ▾

YARN (MR2 Included) (Cluster 1) Actions ▾

Status Instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI ▾ Quick Links ▾

Switch to 1

Filters

SCOPE

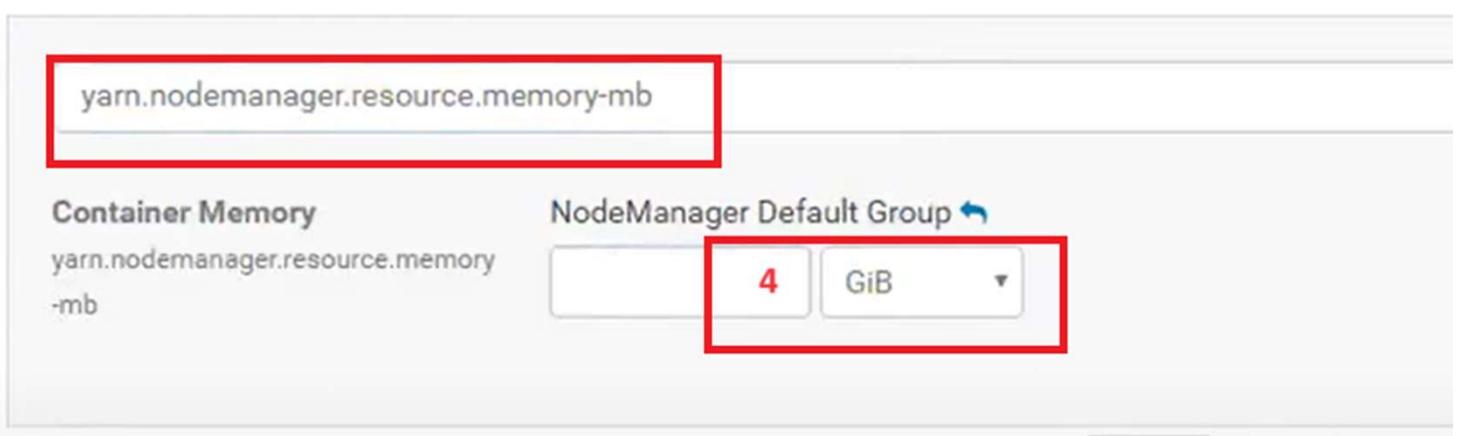
YARN (MR2 Included) (Service-Wide)	0
Gateway	0
JobHistory Server	0
NodeManager	1
ResourceManager	0

yarn.nodemanager.resource.memory-mb

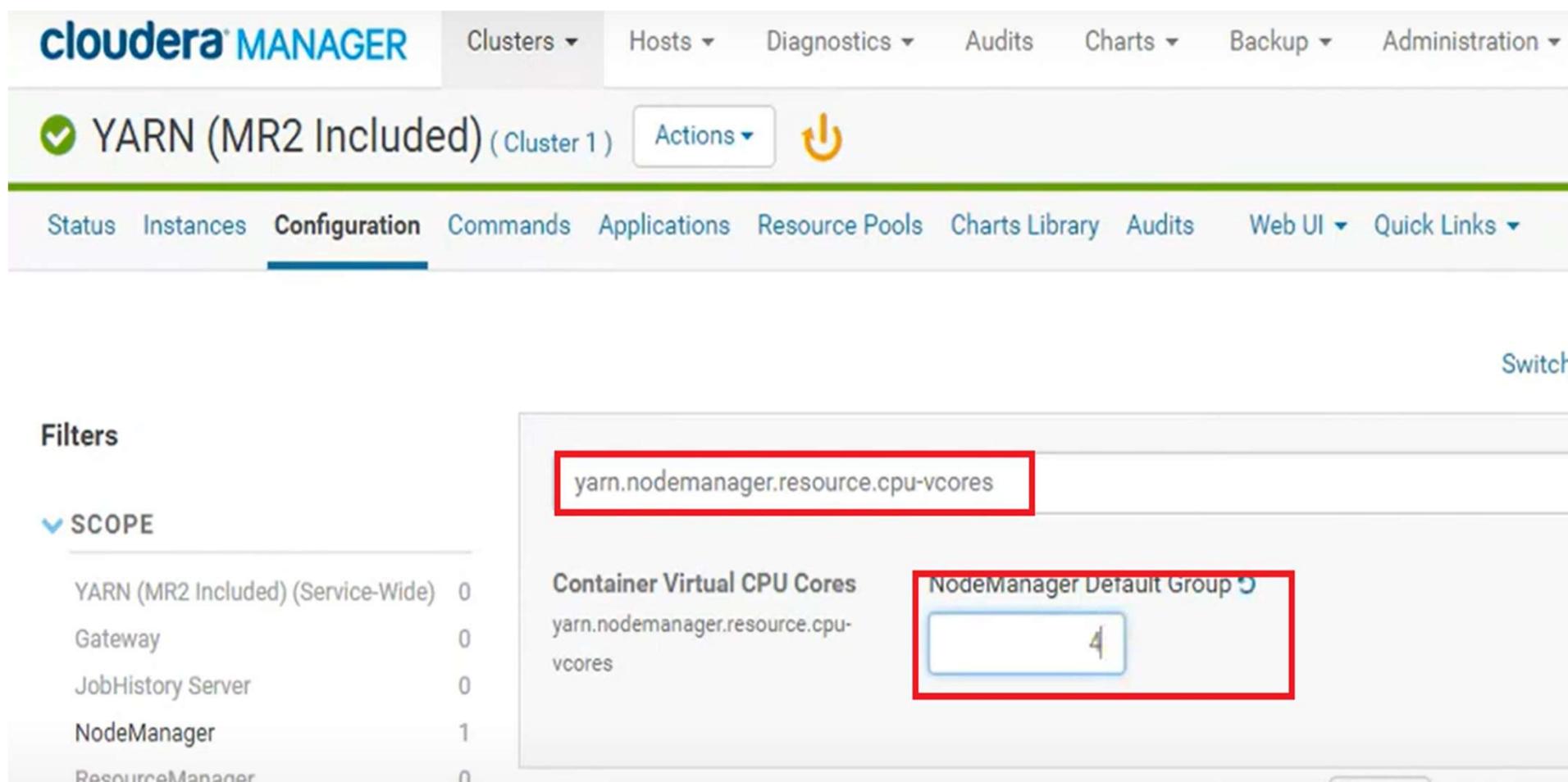
Container Memory NodeManager Default Group ↵

yarn.nodemanager.resource.memory -mb

4 GiB ▾



# CPU Cores Available Update



cloudera MANAGER

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

YARN (MR2 Included) (Cluster 1) Actions ▾ 

Status Instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI ▾ Quick Links ▾

Switch

Filters

SCOPE

Scope	Value
YARN (MR2 Included) (Service-Wide)	0
Gateway	0
JobHistory Server	0
NodeManager	1
ResourceManager	0

yarn.nodemanager.resource.cpu-vcores

Container Virtual CPU Cores

yarn.nodemanager.resource.cpu-vcores

NodeManager Default Group

4

# Few More Details

## STEP 5: Verify YARN Settings on Cluster

Go to the Resource Manager Web UI (usually <http://<ResourceManagerIP>:8088/>) and verify the "Memory Total" and "Vcores Total" matches the values above. If your machine has no bad nodes, then the numbers should match exactly.

Resource Manager Property to Check	Value	Note
Expected Value for "Vcores Total"	12	Calculated from STEP 2 "YARN Available Vcores"
Expected Value for "Memory Total" (in GB)	12	Calculated from STEP 2 "YARN Available Memory (GB)"

## STEP 6: Verify Container Settings on Cluster

In order to have YARN jobs run cleanly, you need to configure the container properties.

YARN Container Configuration Property (Vcores)	Value	Description
yarn.scheduler.minimum-allocation-vcores	1	Minimum vcore reservation for a container
yarn-scheduler.maximum-allocation-vcores	1	Maximum vcore reservation for a container
yarn.scheduler.increment-allocation-vcores	1	Vcore allocations must be a multiple of this value
YARN Container Configuration Property (Memory)	Value	

# Scheduler Info

## STEP 6: Verify Container Settings on Cluster

In order to have YARN jobs run cleanly, you need to configure the container properties.

YARN Container Configuration Property (Vcores)	Value	Description
yarn.scheduler.minimum-allocation-vcores	1	Minimum vcore reservation for a container
yarn.scheduler.maximum-allocation-vcores	1	Maximum vcore reservation for a container
yarn.scheduler.increment-allocation-vcores	1	Vcore allocations must be a multiple of
YARN Container Configuration Property (Memory)	Value	
yarn.scheduler.minimum-allocation-mb	1024	Minimum memory reservation for a container
yarn.scheduler.maximum-allocation-mb	8192	Maximum memory reservation for a container
yarn.scheduler.increment-allocation-mb	512	Memory allocations must be a multiple of

# Check

## 41 STEP 6B: Container Sanity Checking

42 This section will do some basic checking of your container parameters in STEP 6 against the hosts.

### 43

#### 44 Sanity Check

45 Vcore Max >= Vcore Min

46 Memory Max >= Memory Min

47 VCoreMin >= 0

48 VCoreMin <= HostsVCores

49 VCoreMax >= 1

50 VCoreMax <= HostsVcores

51 Memory Min < 1024 MB

52 Memory Max <= HostsMemory

53

54

Means GOOD TO GO

Check Status	Description
GOOD	yarn.scheduler.maximum-allocation-vcores
GOOD	yarn.scheduler.maximum-allocation-mb
GOOD	yarn.scheduler.minimum-allocation-vcores
GOOD	yarn.scheduler.minimum-allocation-mb
GOOD	yarn.scheduler.maximum-allocation-vcores
GOOD	yarn.scheduler.maximum-allocation-mb
GOOD	yarn.scheduler.minimum-allocation-vcores
GOOD	yarn.scheduler.maximum-allocation-mb

# Map-Reduce Config

## MapReduce Configuration

### STEP 7: MapReduce Configuration

Property	Property Type	Component	Value	Description
yarn.app.mapreduce.am.resource.cpu-vcores	Config	Application Master	1	AM cores
yarn.app.mapreduce.am.resource.mb	Config	Application Master	1024	AM memory
ApplicationMaster Java Maximum Heap Size (available in CM)	Java VM Heap	Application Master	1024	AM Java heap
mapreduce.map.cpu.vcores	Config	Map Task	1	Map cores
mapreduce.map.memory.mb	Config	Map Task	1024	Map memory
mapreduce.map.java.opts.max.heap	Java VM Heap	Map Task	1024	Map Java heap
mapreduce.reduce.cpu.vcores	Config	Reduce Task	1	Reduce cores
mapreduce.reduce.memory.mb	Config	Reduce Task	1024	Reduce memory
mapreduce.reduce.java.opts	Java VM Heap	Reduce Task	1024	Reduce Java heap
mapreduce.task.io.sort.mb	Config	Spill/Sort (Map Task)	256	Spill/Sort memory

### STEP 7A: MapReduce Sanity Checking

## Sanity check MapReduce settings against container minimum/maximum properties.

Application Master Sanity Checks				Value	Description		
yarn.app.mapreduce.am.resource.cpu-vcores >= container min				GOOD	Make sure ApplicationMaster vcore request fits within container limits		
yarn.app.mapreduce.am.resource.cpu-vcores <= container max				GOOD	Ditto		
yarn.app.mapreduce.am.resource.mb >= container min				GOOD	Make sure ApplicationMaster memory request fits within container lim		
yarn.app.mapreduce.am.resource.mb <= container max				GOOD	Ditto		
ApplicationMaster Java Heap "close" to memory request				GOOD	Make sure ApplicationMaster Java Heap is within 90% to 100% of yarn.		
Map Task Sanity Checks				Value	Description		
mapreduce.map.cpu.vcores >= container min				GOOD	Make sure Map Task vcore request fits within container limits		
mapreduce.map.cpu.vcores <= container max				GOOD	Ditto		
mapreduce.map.cpu.memory.mb >= container min				GOOD	Make sure Map Task memory request fits within container limits		
mapreduce.map.cpu.memory.mb <= container max				GOOD	Ditto		
Map Task Java Heap "close" to memory request				GOOD	Make sure that Map Task Java Heap is within 90% to 100% of mapredu		
mapreduce.task.io.sort.mb << Map Task Java Heap				GOOD	Make sure that Spill/Sort memory reservation leaves enough "room" in		
Reduce Task Sanity Checks				Value	Description		
mapreduce.reduce.cpu.vcores >= container min				GOOD	Make sure Reduce Task vcore request fits within container limits		
mapreduce.reduce.cpu.vcores <= container max				GOOD	Ditto		
mapreduce.reduce.cpu.memory.mb >= container min				GOOD	Make sure Reduce Task memory request fits within container limits		
mapreduce.reduce.cpu.memory.mb <= container max				GOOD	Ditto		
Reduce Task Java Heap "close" to memory request				GOOD	Make sure that Reduce Task Java Heap is within 90% to 100% of mapre		

# Restart Services

Cloudera MANAGER

Clusters ▾ Hosts ▾ Diagnostics ▾ Audits Charts ▾ Backup ▾ A

## Home

Status All Health Issues Configuration ⚡ 6 All Recent Commands

Cluster 1 (CDH 5.12.1, Parcels)

Hosts 3

HDFS

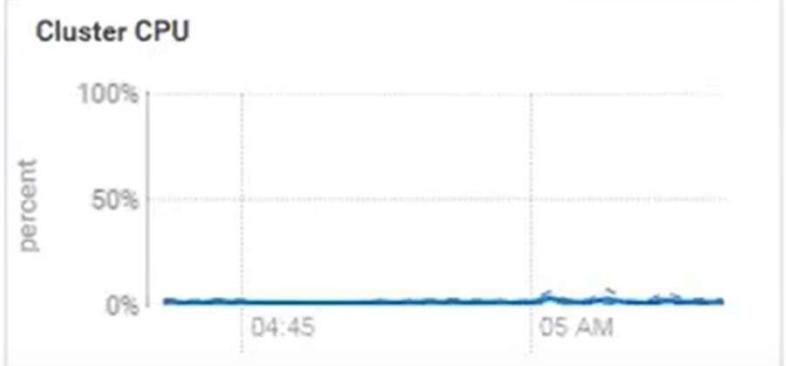
YARN (MR2...)

**Restart Services**

**Stale Configuration: Client configuration redeployment needed**

### Charts

Cluster CPU



Time	Usage (%)
04:45	~5%
05 AM	~5%

# Verify

Logged in as: dr.



## All Applications

### Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved
0	0	0	0	0	0 B	12 GB	0 B	0	12	0

### Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
3	0	0	0	0	0

### User Metrics for dr.who

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	Vcores Used	Vcores Pending	Vcores Reserved
0	0	0	0	0	0	0	0 B	0 B	0 B	0	0	0

Show 20 ▾ entries

Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	Allocated Memory MB	Progress	Tracking UI
----	------	------	------------------	-------	-----------	------------	-------	-------------	--------------------	----------------------	---------------------	----------	-------------