

# Introduction to Cloud Computing

## Qloud Demonstration

15-319, spring 2010

3<sup>rd</sup> Lecture, Jan 19<sup>th</sup>

**Suhail Rehman**

# Time to check out the Qloud!

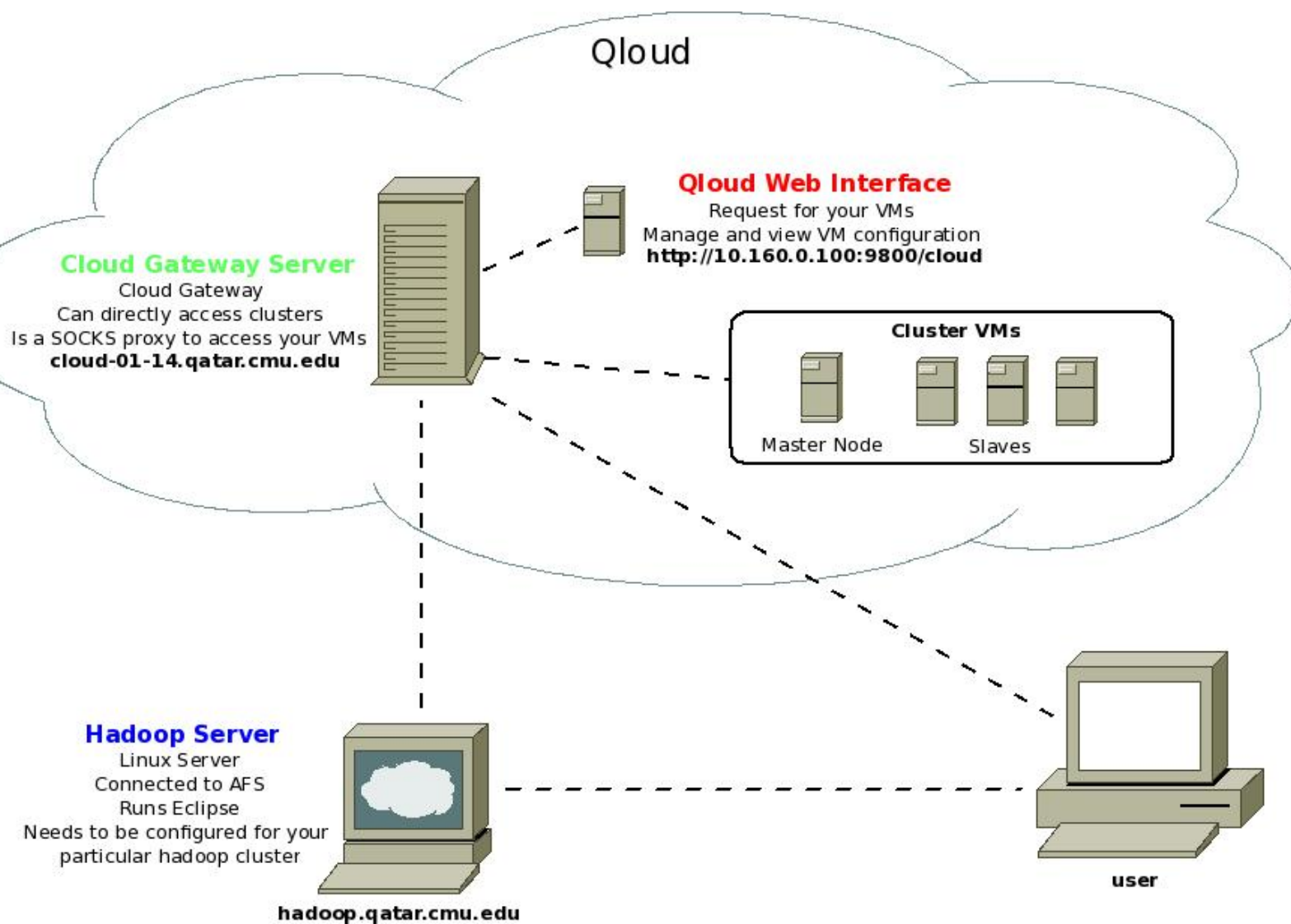
- **Enough Talk!**
- **Time for some Action!**
- **Finally you can have your own Cloud (Virtual Machines)!**
- **Get your own Cloud from Qloud!**

# Time to check out the Qloud!

- Enough Talk!
- Time for some Action!
- Finally you can have your own Cloud (Virtual Machines)!
- Get your own Cloud from Qloud!



# User's Qloud Perspective



# Important Qloud servers and interfaces

## ■ Hadoop Server

[hadoop.qatar.cmu.edu](http://hadoop.qatar.cmu.edu)

- User workspace (Hadoop/Eclipse)
- AFS access and login

## ■ Cloud Gateway Server

[cloud-01-14.qatar.cmu.edu](http://cloud-01-14.qatar.cmu.edu)

- Gives you access the virtualized resources of the cloud
- Will be a SOCKS proxy for all your Cloud and Hadoop tasks

## ■ Qloud Web Interface

<http://10.160.0.100:9080/cloud/>

- Easy web interface to request your Cloud
- Once provisioned, you can checkout the vital stats of your cloud

# Steps to get your own Cloud

- Set the **Cloud Gateway Server** as the SOCKS proxy in your Browser
- Log on to the **Qloud Web Interface** and request your Cloud
- Wait for our uber-geek (aka Brian) to approve
- Once Brian approves it, you'll have your cloud in 2 hours
- The entire process should take less than 24 hours 😊
  - You cannot request a cloud at 2am and expect it to be ready at 4am

# Qloud Web Interface

## Cloud Computing Center



[Home](#) [Reports](#) [Manage Cloud Users](#)

Welcome rehman [ [logout](#) ]

### Project Summary

- 0 New**
- 2 Active**
- 3 Completed
- 0 Failed**

### Upcoming Events

- Jan 26, 2010 Final Hadoop Test will be **deprovisioned**
- Jan 31, 2010 New Hadoop Testbed will be **deprovisioned**

### My Projects

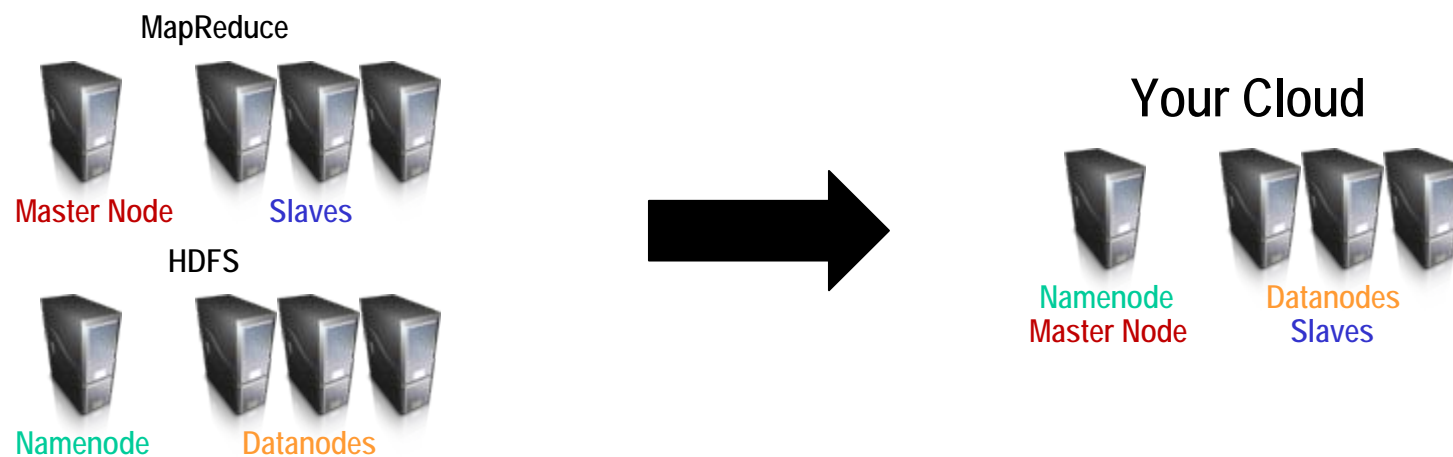
- |                           |                                |                     |  |
|---------------------------|--------------------------------|---------------------|--|
| <b>Final Hadoop Test</b>  | 4 active servers (4 requested) | 1/12/10 to 1/26/10  | Approved <b>Active</b> ●                 |
| <b>New Hadoop Testbed</b> | 4 active servers (4 requested) | 12/28/09 to 1/31/10 | Approved <b>Active</b> ● (Dates Changed) |

Click on a project to see details and request changes

[Refresh](#) | [Request New Cloud Project](#)

# It is time for Hadoop ☺

- The Hadoop infrastructure allows you to run map-reduce jobs distributed over your virtual machines
- In Hadoop MapReduce, one node is designated as the **Master Node**, and the rest are **slaves**.
- HDFS requires one **Namenode** and several **Datanodes**.
- In our setup, the **Master Node** and **Namenode** are the same machine.





# Hadoop on Your Cloud

## Cloud Computing Center



Home

Welcome rehman [logout]

### Project Details

Project Name **Final Hadoop Test**Customer **Suhail\_Rehman**Project Type **Hadoop customized for CMU**Description **Final test before Project 1 goes live.**Project State **Active**Requested Server Count **4**Active Server Count **4**Start Date **Jan 12, 2010**End Date **Jan 26, 2010**Duration **14 days**

Project Approved

### Project Infrastructure

Name	Hardware Configuration	Base Image	Status
<a href="#">vm-10-160-2-29</a>	1.0CPU (1 vcpu) - 1024MB Memory - 80GB Disk (incl. 2048MB swap)	Xen RedHat Linux 5.2	Active
<a href="#">vm-10-160-2-28</a>	1.0CPU (1 vcpu) - 1024MB Memory - 80GB Disk (incl. 2048MB swap)	Xen RedHat Linux 5.2	Active
<a href="#">vm-10-160-2-27</a>	1.0CPU (1 vcpu) - 1024MB Memory - 80GB Disk (incl. 2048MB swap)	Xen RedHat Linux 5.2	Active
<a href="#">vm-10-160-2-26</a>	3.0CPU (3 vcpus) - 1024MB Memory - 60GB Disk (incl. 2048MB swap)	Xen RedHat Linux 5.2	Active

System Info		Additional Software	Real Time Monitoring	Remote Control
IP	<b>10.160.2.26</b>	Hadoop 0.20.1	<b>server offline</b>	Power On
OS Type	<b>Xen RedHat Linux 5.2</b>	IBM Java SDK 6		Power Off
Pool / Type	<b>Xen System x (xen)</b>			Restart
Admin Password	<b>DN30QNLZ</b>			

Slaves  
and  
Datanodes

Master Node and Namenode

# Where to go from here?

## ■ Logon to your Master Node

- ssh to [cloud-01-14.qatar.cmu.edu](https://cloud-01-14.qatar.cmu.edu) and then ssh to your **master node**

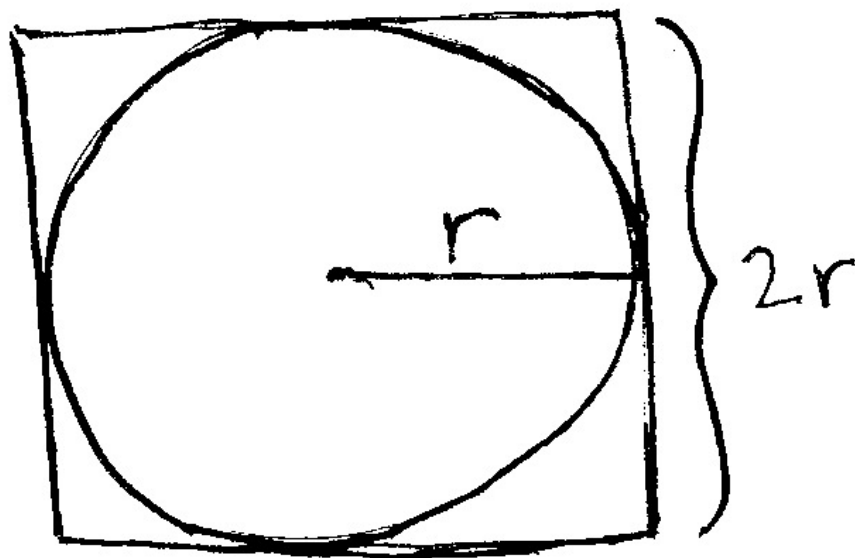
## ■ Setup Hadoop

- Fortunately, your VM's automatically have the correct configuration files for Hadoop the moment they are provisioned (Thanks to Brian!)
- All you need to do is format HDFS and start the Hadoop services.

## ■ Lets try running some sample code on Hadoop

# Sample MapReduce Code- Estimate $\pi$

- Estimating  $\pi$  by random sampling
- Imagine you have a dart board like so:



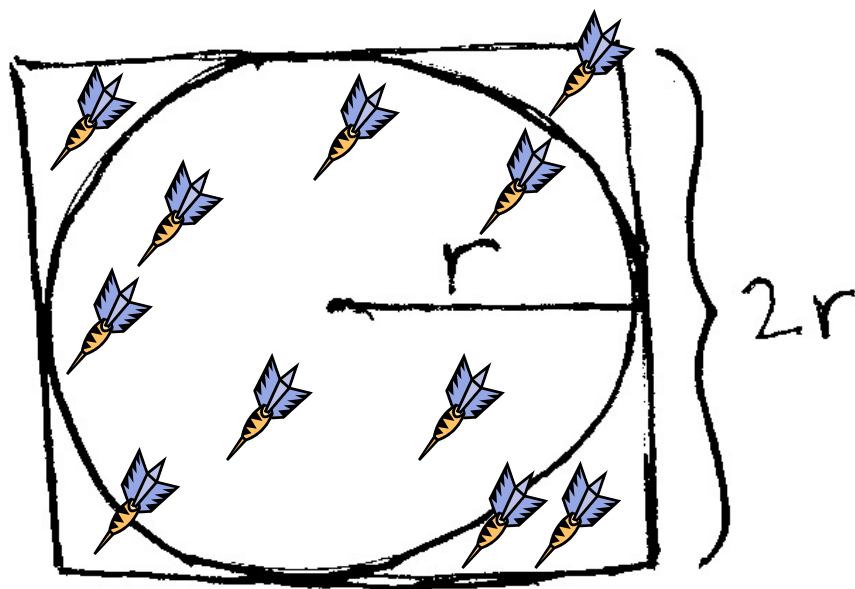
$$P(\text{dart in circle}) = \frac{\pi r^2}{4r^2} = \frac{\pi}{4}$$

$$\Rightarrow \pi = 4P(\text{dart in circle})$$

- $\pi$  is simply the (ratio of darts that land inside the circle to the total number of darts thrown) times 4

# Writing this as a Serial Program

- Throw  $N$  darts on the board. Each dart lands at a random position  $(x,y)$  on the board.



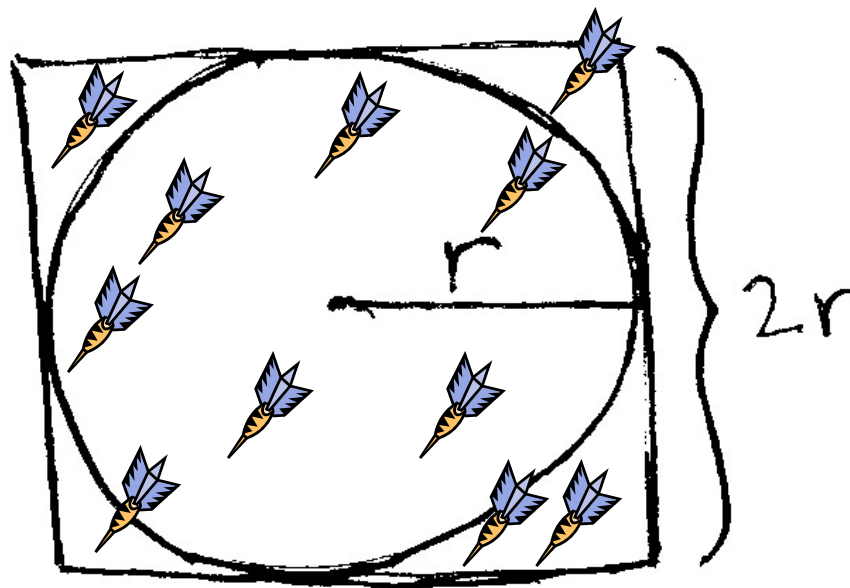
- Note if each dart landed inside the circle or not
  - Check if  $x^2 + y^2 < r$
- Take the total number of darts that landed in the circle as  $S$

$$4 \left( \frac{S}{N} \right) = \pi$$

# But I have Millions of Darts!

- If you want to get an accurate estimate of Pi, you need a large number of random samples.
- Notice that each dart can be thrown at any time and its position can be evaluated independently
- With one person throwing all the darts, it will take a long time to finish
- If we had N people throwing a dart each, this would be much faster!

# But I have Millions of Darts!



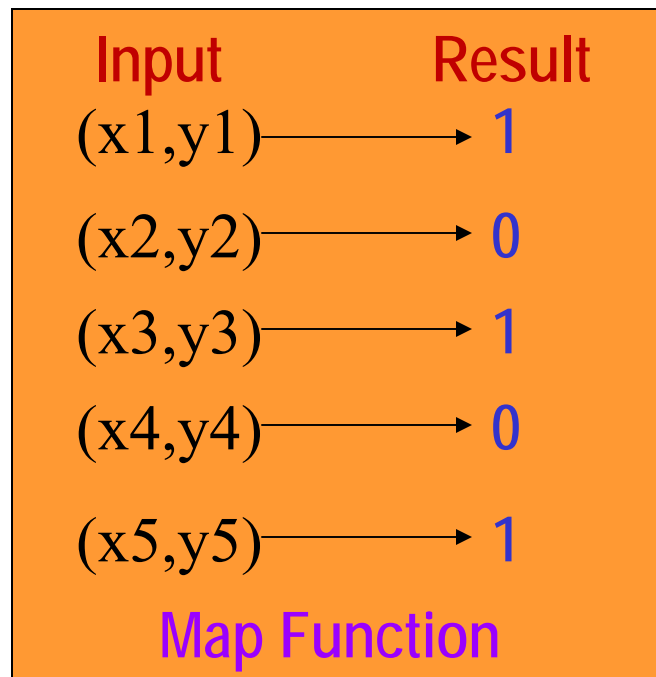
# How do you do this in Parallel?

- Let  $(x,y)$  be a random position of the dart inside the square.
- Each  $(x,y)$  pair can be evaluated independently.
- Let us “map” each  $(x,y)$  pair to a result – the result being whether it is inside the circle (**1**) or not (**0**).

Input	Result
$(x_1, y_1)$	1
$(x_2, y_2)$	0
$(x_3, y_3)$	1
$(x_4, y_4)$	0
$(x_5, y_5)$	1

# The **Map** function

- A Map function takes input values and produces an output for each input value in parallel.





# ....and then?

- So we have results of each  $(x,y)$  pair – lots of them
- We need to find the number of points inside the circle.  
We need to sum up the values

Result

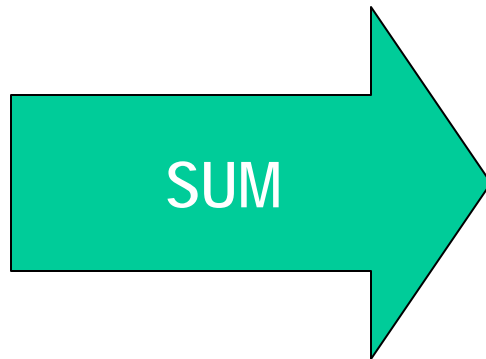
1

0

1

0

1



**S**

# The Reduce Function

- A Reduce function takes input values from the Map functions and produces output using a user defined operation.

Result

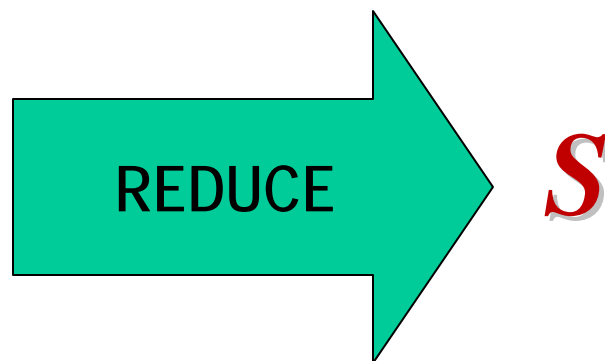
1

0

1

0

1



- In this case, addition is the reduce operation.

# What about Pi?

- Now that we have the total number of points inside the circle,  $S$  and the total number of points  $N$  we've sampled...

$$4 \left( \frac{S}{N} \right) = \pi^*$$

\*Subject to Terms and Conditions

1.  $N$  should be large
2. Points should be chosen uniformly at random

# Running PI **MapReduce** Code

- The MapReduce code creates random  $(x,y)$  pair values
- It gives each node a number of  $(x,y)$  pairs and evaluates if it's in the circle or not (**MAP**)
- Then some nodes will collect the results of these samples, evaluate the percentage and calculate  $\pi$  (**REDUCE**)
- Running the hadoop example:

```

hadoop jar hadoop-0.20.1-examples.jar pi 10 100

```

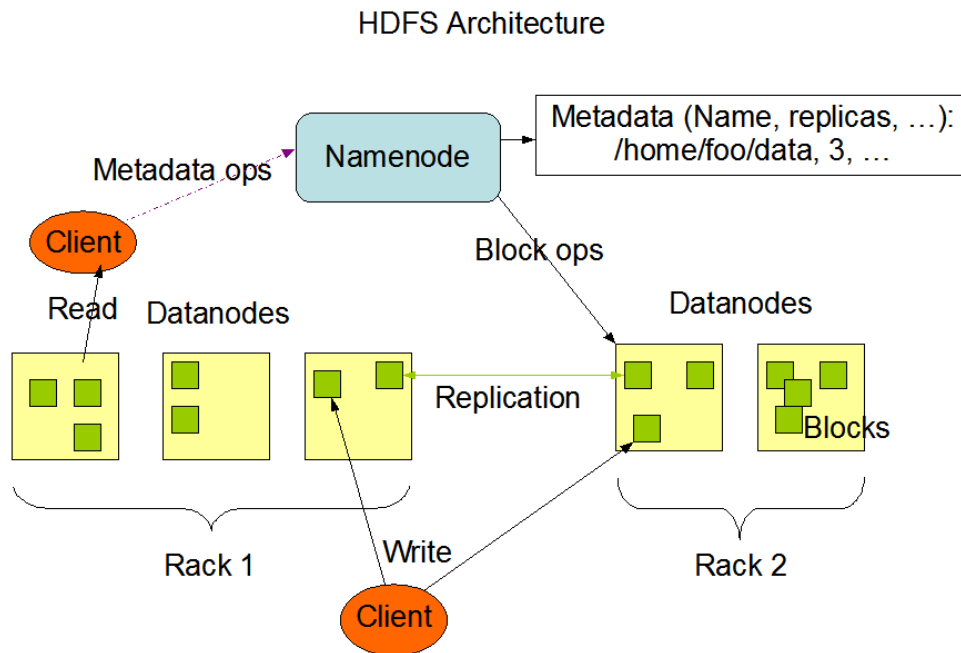
Run a jar file      The Jar file      Name of the java class      #maps      #samples per map

# Working with Files in Hadoop

- Notice that the Pi example randomly generates input, it does not require any user files.
- Hadoop is mainly used to work with large data, and large data is always in a file.
- HDFS to the rescue!

# HDFS Basics

- HDFS is the Hadoop Distributed File System.
- Files are distributed over all four nodes and are triple-replicated, by default, to tolerate failure.



# HDFS Commands

- All commands begin with `hadoop dfs`

UNIX command	Hadoop HDFS Command
<code>ls /</code>	<code>hadoop dfs -ls /</code>
<code>cat /dir/filename</code>	<code>hadoop dfs -cat /dir/filename</code>
<code>mkdir dir1</code>	<code>hadoop dfs -mkdir /dir1</code>
<code>rm /dir/filename</code>	<code>hadoop dfs -rm /dir/filename</code>
<code>rm -r /dir</code>	<code>hadoop dfs -rmr /dir</code>

# Handling Files in HDFS

## ■ To add files to HDFS:

- `hadoop dfs -put localfilename /hdfs_dir/remotefilename`

## ■ To copy files from HDFS to local filesystem

- `hadoop dfs -get /hdfs_dir/remotefilename localfilename`

## ■ To copy files inside HDFS filesystem

- `hadoop dfs -cp /hdfs_dir/file1 /hdfs_dir/file2`



# Keeping track of your Hadoop & HDFS

- **Hadoop MapReduce has a JobTracker web interface**
  - Keeps Track of the submitted jobs, time taken, errors, logs etc.
  - [http://MASTER\\_NODE\\_IP:50030](http://MASTER_NODE_IP:50030)
  
- **The HDFS Namenode also maintains a web interface**
  - Browse your HDFS files
  - See how much disk space you have remaining in your HDFS.
  - [http://NAME\\_NODE\\_IP:50070](http://NAME_NODE_IP:50070)

# Setting up Eclipse

- Might be easier to work with an IDE when developing large applications in Hadoop.
- Eclipse is available on [hadoop.qatar.cmu.edu](http://hadoop.qatar.cmu.edu) with the MapReduce plugin
- Setup and Run eclipse @ [hadoop.qatar.cmu.edu](http://hadoop.qatar.cmu.edu)
  - Use xwin32 on windows machines to run eclipse remotely
  - Configure Eclipse to use your cloud
  - Start developing MapReduce applications