Hadoop 2.6 Installing on Ubuntu 14.04 (Single-Node Cluster)

Hadoop on Ubuntu 14.04

In this chapter, we'll install a single-node Hadoop cluster backed by the Hadoop Distributed File System on Ubuntu.

Installing Java

Hadoop framework is written in Java!!

```
k@laptop:~$ cd ~

# Update the source list
k@laptop:~$ sudo apt-get update

# The OpenJDK project is the default version of J
# that is provided from a supported Ubuntu reposi
k@laptop:~$ sudo apt-get install default-jdk

k@laptop:~$ java -version
java version "1.7.0_65"
OpenJDK Runtime Environment (IcedTea 2.5.3) (7u71
OpenJDK 64-Bit Server VM (build 24.65-b04, mixed
```

Adding a dedicated Hadoop user

```
k@laptop:~$ sudo addgroup hadoop
Adding group `hadoop' (GID 1002) ...
Done.

k@laptop:~$ sudo adduser --ingroup hadoop hduser
Adding user `hduser' ...
Adding new user `hduser' (1001) with group `hadoo
Creating home directory `/home/hduser' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for hduser
Enter the new value, or press ENTER for the defau
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
```

```
        Other []:
Is the information correct? [Y/n] Y
```

Installing SSH

ssh has two main components:

1. **ssh** : The command we use to connect to remote machines - the client.
2. **sshd** : The daemon that is running on the server and allows clients to connect to the server.

The **ssh** is pre-enabled on Linux, but in order to start **sshd** daemon, we need to install **ssh** first. Use this command to do that :

```
k@laptop:~$ sudo apt-get install ssh
```

This will install ssh on our machine. If we get something similar to the following, we can think it is setup properly:

```
k@laptop:~$ which ssh
/usr/bin/ssh

k@laptop:~$ which sshd
/usr/sbin/sshd
```

Create and Setup SSH Certificates

Hadoop requires SSH access to manage its nodes, i.e. remote machines plus our local machine. For our single-node setup of Hadoop, we therefore need to configure SSH access to localhost.

So, we need to have SSH up and running on our machine and configured it to allow SSH public key authentication.

Hadoop uses SSH (to access its nodes) which would normally require the user to enter a password. However, this requirement can be eliminated by creating and setting up SSH certificates using the following commands. If asked for a filename just leave it blank and press the enter key to continue.

```
k@laptop:~$ su hduser
Password:
k@laptop:~$ ssh-keygen -t rsa -P ""
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hduser
Created directory '/home/hduser/.ssh'.
Your identification has been saved in /home/hduse
Your public key has been saved in /home/hduser/.s
The key fingerprint is:
50:6b:f3:fc:0f:32:bf:30:79:c2:41:71:26:cc:7d:e3 h
The key's randomart image is:
+--[ RSA 2048]----+
|        .oo.o    |
|       . .o=. o  |
|      . + .  o . |
|       o =    E  |
|        S +      |
|         . +     |
|          O +    |
|           O o   |
|            o..  |
+-----------------+


hduser@laptop:/home/k$ cat $HOME/.ssh/id_rsa.pub
```

The second command adds the newly created key to the list of authorized keys so that Hadoop can use ssh without prompting for a password.

We can check if ssh works:

```
hduser@laptop:/home/k$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)'
ECDSA key fingerprint is e1:8b:a0:a5:75:ef:f4:b4:
Are you sure you want to continue connecting (yes
Warning: Permanently added 'localhost' (ECDSA) to
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-4
...
```

Apache Hadoop : Creating HBase table with HBase shell and HUE

Apache Hadoop : Creating HBase table with Java API

Apache Hadoop : HBase in Pseudo-Distributed mode

Apache Hadoop HBase : Map, Persistent, Sparse, Sorted, Distributed and Multidimensional

Apache Hadoop - Flume with CDH5: a single-node Flume deployment (telnet example)

Apache Hadoop (CDH 5) Flume with VirtualBox : syslog example via NettyAvroRpcClient

List of Apache Hadoop hdfs commands

Apache Hadoop : Creating Wordcount Java Project with Eclipse Part 1

Apache Hadoop : Creating Wordcount Java Project with Eclipse Part 2

Apache Hadoop : Creating Card Java Project with Eclipse using Cloudera VM

Install Hadoop

```
hduser@laptop:~$ wget http://mirrors.sonic.net/ap
hduser@laptop:~$ tar xvzf hadoop-2.6.0.tar.gz
```

We want to move the Hadoop installation to the **/usr/local/hadoop** directory using the following command:

```
hduser@laptop:~/hadoop-2.6.0$ sudo mv * /usr/loca
[sudo] password for hduser:
hduser is not in the sudoers file.  This incident
```

Oops!... We got:

```
"hduser is not in the sudoers file. This incident
```

This error can be resolved by logging in as a root user, and then add **hduser** to **sudo**:

```
hduser@laptop:~/hadoop-2.6.0$ su k
Password:

k@laptop:/home/hduser$ sudo adduser hduser sudo
[sudo] password for k:
Adding user `hduser' to group `sudo' ...
Adding user hduser to group sudo
Done.
```

Now, the **hduser** has root priviledge, we can move the Hadoop installation to the **/usr/local/hadoop** directory without any problem:

```
k@laptop:/home/hduser$ sudo su hduser

hduser@laptop:~/hadoop-2.6.0$ sudo mv * /usr/loca
hduser@laptop:~/hadoop-2.6.0$ sudo chown -R hduse
```

UnoExample for CDH5 - local run

Apache Hadoop : Creating Wordcount Maven Project with Eclipse

Wordcount MapReduce with Oozie workflow with Hue browser - CDH 5.3 Hadoop cluster using VirtualBox and QuickStart VM

Spark 1.2 using VirtualBox and QuickStart VM - wordcount

Spark Programming Model : Resilient Distributed Dataset (RDD)

Apache Spark 1.3 with PySpark (Spark Python API) Shell

Apache Spark 1.2 with PySpark (Spark Python API) Wordcount using CDH5

Apache Spark 1.2 Streaming

Setup Configuration Files

The following files will have to be modified to complete the Hadoop setup:

1. ~/.bashrc
2. /usr/local/hadoop/etc/hadoop/hadoop-env.sh
3. /usr/local/hadoop/etc/hadoop/core-site.xml
4. /usr/local/hadoop/etc/hadoop/mapred-site.xml.template
5. /usr/local/hadoop/etc/hadoop/hdfs-site.xml

1. ~/.bashrc:

Before editing the **.bashrc** file in our home directory, we need to find the path where Java has been installed to set the **JAVA_HOME** environment variable using the following command:

```
hduser@laptop update-alternatives --config java
There is only one alternative in link group java
Nothing to configure.
```

Now we can append the following to the end of **~/.bashrc**:

```
hduser@laptop:~$ vi ~/.bashrc

#HADOOP VARIABLES START
```

```
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd6
export HADOOP_INSTALL=/usr/local/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTA
export HADOOP_OPTS="-Djava.library.path=$HADOOP_I
#HADOOP VARIABLES END

hduser@laptop:~$ source ~/.bashrc
```

note that the JAVA_HOME should be set as the path just before the '.../bin/':

```
hduser@ubuntu-VirtualBox:~$ javac -version
javac 1.7.0_75

hduser@ubuntu-VirtualBox:~$ which javac
/usr/bin/javac

hduser@ubuntu-VirtualBox:~$ readlink -f /usr/bin/
/usr/lib/jvm/java-7-openjdk-amd64/bin/javac
```

2. /usr/local/hadoop/etc/hadoop/hadoop-env.sh

We need to set **JAVA_HOME** by modifying **hadoop-env.sh** file.

```
hduser@laptop:~$ vi /usr/local/hadoop/etc/hadoop/

export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd6
```

Adding the above statement in the **hadoop-env.sh** file ensures that the value of JAVA_HOME variable will be available to Hadoop whenever it is started up.

3. /usr/local/hadoop/etc/hadoop/core-site.xml:

D3.js

DevOps

Phases of Continuous Integration

Software development methodology

Introduction to DevOps

Samples of Continuous Integration (CI) / Continuous Delivery (CD) - Use cases

Artifact repository and repository management

Linux - General, shell programming, processes & signals ...

RabbitMQ...

MariaDB

New Relic APM with NodeJS : simple agent setup on AWS instance

Nagios - The industry standard in IT infrastructure monitoring on Ubuntu

DevOps / Sys Admin Q & A

The **/usr/local/hadoop/etc/hadoop/core-site.xml** file contains configuration properties that Hadoop uses when starting up.
This file can be used to override the default settings that Hadoop starts with.

```
hduser@laptop:~$ sudo mkdir -p /app/hadoop/tmp
hduser@laptop:~$ sudo chown hduser:hadoop /app/ha
```

Open the file and enter the following in between the <configuration></configuration> tag:

```
hduser@laptop:~$ vi /usr/local/hadoop/etc/hadoop/

<configuration>
 <property>
  <name>hadoop.tmp.dir</name>
  <value>/app/hadoop/tmp</value>
  <description>A base for other temporary directo
 </property>

 <property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:54310</value>
  <description>The name of the default file syste
  scheme and authority determine the FileSystem i
  uri's scheme determines the config property (fs
  the FileSystem implementation class.  The uri's
  determine the host, port, etc. for a filesystem
 </property>
</configuration>
```

4. /usr/local/hadoop/etc/hadoop/mapred-site.xml

By default, the **/usr/local/hadoop/etc/hadoop/** folder contains
/usr/local/hadoop/etc/hadoop/mapred-site.xml.template file which has to be renamed/copied with the name **mapred-site.xml**:

```
hduser@laptop:~$ cp /usr/local/hadoop/etc/hadoop/
```

The **mapred-site.xml** file is used to specify which

framework is being used for MapReduce.
We need to enter the following content in between the <configuration></configuration> tag:

```
<configuration>
 <property>
  <name>mapred.job.tracker</name>
  <value>localhost:54311</value>
  <description>The host and port that the MapRedu
  at.  If "local", then jobs are run in-process a
  and reduce task.
  </description>
 </property>
</configuration>
```

5. /usr/local/hadoop/etc/hadoop/hdfs-site.xml

The **/usr/local/hadoop/etc/hadoop/hdfs-site.xml** file needs to be configured for each host in the cluster that is being used.
It is used to specify the directories which will be used as the **namenode** and the **datanode** on that host.

Before editing this file, we need to create two directories which will contain the namenode and the datanode for this Hadoop installation.
This can be done using the following commands:

```
hduser@laptop:~$ sudo mkdir -p /usr/local/hadoop_
hduser@laptop:~$ sudo mkdir -p /usr/local/hadoop_
hduser@laptop:~$ sudo chown -R hduser:hadoop /usr
```

Open the file and enter the following content in between the <configuration></configuration> tag:

```
hduser@laptop:~$ vi /usr/local/hadoop/etc/hadoop/

<configuration>
 <property>
  <name>dfs.replication</name>
  <value>1</value>
```

(16B) - Serving multiple domains using server block - Nginx

(17) - Linux startup process

Jenkins

Install

Configuration - Manage Jenkins - security setup

Adding job and build

Scheduling jobs

Managing_plugins

Git/GitHub plugins, SSH keys configuration, and Fork/Clone

JDK & Maven setup

Build configuration for GitHub Java application with Maven

Build Action for GitHub Java application with Maven - Console Output, Updating Maven

Commit to changes to GitHub & new test results - Build Failure

Commit to changes to GitHub & new test results

```
  <description>Default block replication.
  The actual number of replications can be specif
  The default is used if replication is not speci
  </description>
 </property>
 <property>
   <name>dfs.namenode.name.dir</name>
   <value>file:/usr/local/hadoop_store/hdfs/namen
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/usr/local/hadoop_store/hdfs/datan
 </property>
</configuration>
```


Format the New Hadoop Filesystem

Now, the Hadoop file system needs to be formatted so that we can start to use it. The format command should be issued with write permission since it creates **current** directory
under **/usr/local/hadoop_store/hdfs/namenode** folder:

```
hduser@laptop:~$ hadoop namenode -format
DEPRECATED: Use of this script to execute hdfs co
Instead use the hdfs command for it.

15/04/18 14:43:03 INFO namenode.NameNode: STARTUP
/************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = laptop/192.168.1.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 2.6.0
```

- Successful Build

Adding code coverage and metrics

Jenkins on EC2 - creating an EC2 account, ssh to EC2, and install Apache server

Jenkins on EC2 - setting up Jenkins account, plugins, and Configure System (JAVA_HOME, MAVEN_HOME, notification email)

Jenkins on EC2 - Creating a Maven project

Jenkins on EC2 - Configuring GitHub Hook and Notification service to Jenkins server for any changes to the repository

Jenkins on EC2 - Line Coverage with JaCoCo plugin

Jenkins Build Pipeline & Dependency Graph Plugins

Jenkins Build Flow Plugin

Puppet

Puppet with Amazon AWS I - Puppet accounts

```
STARTUP_MSG:   classpath = /usr/local/hadoop/etc/
...
STARTUP_MSG:   java = 1.7.0_65
************************************************
15/04/18 14:43:03 INFO namenode.NameNode: registe
15/04/18 14:43:03 INFO namenode.NameNode: createN
15/04/18 14:43:07 WARN util.NativeCodeLoader: Una
Formatting using clusterid: CID-e2f515ac-33da-45b
15/04/18 14:43:09 INFO namenode.FSNamesystem: No
15/04/18 14:43:09 INFO namenode.FSNamesystem: fsL
15/04/18 14:43:10 INFO blockmanagement.DatanodeMa
15/04/18 14:43:10 INFO blockmanagement.DatanodeMa
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO util.GSet: Computing capac
15/04/18 14:43:10 INFO util.GSet: VM type       =
15/04/18 14:43:10 INFO util.GSet: 2.0% max memory
15/04/18 14:43:10 INFO util.GSet: capacity      =
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO blockmanagement.BlockManag
15/04/18 14:43:10 INFO namenode.FSNamesystem: fsO
15/04/18 14:43:10 INFO namenode.FSNamesystem: sup
15/04/18 14:43:10 INFO namenode.FSNamesystem: isP
15/04/18 14:43:10 INFO namenode.FSNamesystem: HA
15/04/18 14:43:10 INFO namenode.FSNamesystem: App
15/04/18 14:43:11 INFO util.GSet: Computing capac
15/04/18 14:43:11 INFO util.GSet: VM type       =
15/04/18 14:43:11 INFO util.GSet: 1.0% max memory
15/04/18 14:43:11 INFO util.GSet: capacity      =
15/04/18 14:43:11 INFO namenode.NameNode: Caching
15/04/18 14:43:11 INFO util.GSet: Computing capac
15/04/18 14:43:11 INFO util.GSet: VM type       =
15/04/18 14:43:11 INFO util.GSet: 0.25% max memor
15/04/18 14:43:11 INFO util.GSet: capacity      =
15/04/18 14:43:11 INFO namenode.FSNamesystem: dfs
15/04/18 14:43:11 INFO namenode.FSNamesystem: dfs
15/04/18 14:43:11 INFO namenode.FSNamesystem: dfs
15/04/18 14:43:11 INFO namenode.FSNamesystem: Ret
15/04/18 14:43:11 INFO namenode.FSNamesystem: Ret
15/04/18 14:43:11 INFO util.GSet: Computing capac
15/04/18 14:43:11 INFO util.GSet: VM type       =
15/04/18 14:43:11 INFO util.GSet: 0.0299999993294
15/04/18 14:43:11 INFO util.GSet: capacity      =
15/04/18 14:43:11 INFO namenode.NNConf: ACLs enab
15/04/18 14:43:11 INFO namenode.NNConf: XAttrs en
15/04/18 14:43:11 INFO namenode.NNConf: Maximum s
15/04/18 14:43:12 INFO namenode.FSImage: Allocate
15/04/18 14:43:12 INFO common.Storage: Storage di
15/04/18 14:43:12 INFO namenode.NNStorageRetentio
15/04/18 14:43:12 INFO util.ExitUtil: Exiting wit
15/04/18 14:43:12 INFO namenode.NameNode: SHUTDOW
/************************************************
```

Puppet with Amazon AWS II (ssh & puppetmaster/puppet install)

Puppet with Amazon AWS III - Puppet running Hello World

Puppet Code Basics - Terminology

Puppet with Amazon AWS on CentOS 7 (I) - Master setup on EC2

Puppet with Amazon AWS on CentOS 7 (II) - Configuring a Puppet Master Server with Passenger and Apache

Puppet master /agent ubuntu 14.04 install on EC2 nodes

Puppet master post install tasks - master's names and certificates setup,

Puppet agent post install tasks - configure agent, hostnames, and sign request

EC2 Puppet master/agent basic tasks - main manifest with a file resource/module and immediate

```
SHUTDOWN_MSG: Shutting down NameNode at laptop/19
************************************************
```

Note that **hadoop namenode -format** command should be executed once before we start using Hadoop.
If this command is executed again after Hadoop has been used, it'll destroy all the data on the Hadoop file system.

Starting Hadoop

Now it's time to start the newly installed single node cluster.
We can use **start-all.sh** or (**start-dfs.sh** and **start-yarn.sh**)

```
k@laptop:~$ cd /usr/local/hadoop/sbin

k@laptop:/usr/local/hadoop/sbin$ ls
distribute-exclude.sh    start-all.cmd        sto
hadoop-daemon.sh         start-all.sh         sto
hadoop-daemons.sh        start-balancer.sh    sto
hdfs-config.cmd          start-dfs.cmd        sto
hdfs-config.sh           start-dfs.sh         sto
httpfs.sh                start-secure-dns.sh  sto
kms.sh                   start-yarn.cmd       yar
mr-jobhistory-daemon.sh  start-yarn.sh        yar
refresh-namenodes.sh     stop-all.cmd
slaves.sh                stop-all.sh

k@laptop:/usr/local/hadoop/sbin$ sudo su hduser

hduser@laptop:/usr/local/hadoop/sbin$ start-all.s
hduser@laptop:~$ start-all.sh
This script is Deprecated. Instead use start-dfs.
15/04/18 16:43:13 WARN util.NativeCodeLoader: Una
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/loc
localhost: starting datanode, logging to /usr/loc
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /
15/04/18 16:43:58 WARN util.NativeCodeLoader: Una
starting yarn daemons
starting resourcemanager, logging to /usr/local/h
localhost: starting nodemanager, logging to /usr/
```

We can check if it's really up and running:

```
hduser@laptop:/usr/local/hadoop/sbin$ jps
9026 NodeManager
7348 NameNode
9766 Jps
8887 ResourceManager
7507 DataNode
```

The output means that we now have a functional instance of Hadoop running on our VPS (Virtual private server).

Another way to check is using **netstat**:

```
hduser@laptop:~$ netstat -plten | grep java
(Not all processes could be identified, non-owned
 will not be shown, you would have to be root to
tcp        0      0 0.0.0.0:50020           0.0.0
tcp        0      0 127.0.0.1:54310         0.0.0
tcp        0      0 0.0.0.0:50090           0.0.0
tcp        0      0 0.0.0.0:50070           0.0.0
tcp        0      0 0.0.0.0:50010           0.0.0
tcp        0      0 0.0.0.0:50075           0.0.0
tcp6       0      0 :::8040                 :::*
tcp6       0      0 :::8042                 :::*
tcp6       0      0 :::8088                 :::*
tcp6       0      0 :::49630                :::*
tcp6       0      0 :::8030                 :::*
tcp6       0      0 :::8031                 :::*
tcp6       0      0 :::8032                 :::*
tcp6       0      0 :::8033                 :::*
```

Stopping Hadoop

```
$ pwd
/usr/local/hadoop/sbin

$ ls
distribute-exclude.sh  httpfs.sh                s
hadoop-daemon.sh       mr-jobhistory-daemon.sh  s
hadoop-daemons.sh      refresh-namenodes.sh     s
hdfs-config.cmd        slaves.sh                s
hdfs-config.sh         start-all.cmd            s
```

We run **stop-all.sh** or (**stop-dfs.sh** and **stop-yarn.sh**) to

Changes

Puppet --configprint

Puppet with Docker

Chef

What is Chef?

Chef install on Ubuntu 14.04 - Local Workstation via omnibus installer

Setting up Hosted Chef server

VirtualBox via Vagrant with Chef client provision

Creating and using cookbooks on a VirtualBox node

Chef server install on Ubuntu 14.04

Chef workstation setup on EC2 Ubuntu 14.04

Chef Client Node - Knife Bootstrapping a node on EC2 ubuntu 14.04

Docker 1.7

Docker install on Amazon Linux AMI

Docker install on EC2 Ubuntu 14.04

stop all the daemons running on our machine:

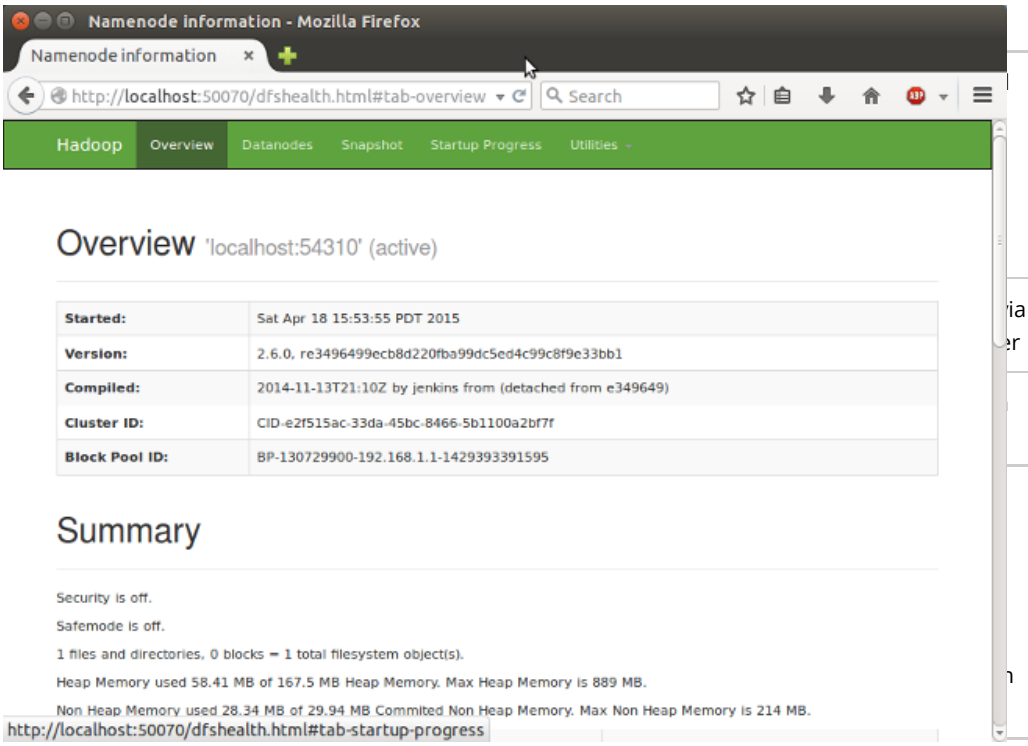
```
hduser@laptop:/usr/local/hadoop/sbin$ pwd
/usr/local/hadoop/sbin
hduser@laptop:/usr/local/hadoop/sbin$ ls
distribute-exclude.sh  httpfs.sh                s
hadoop-daemon.sh       kms.sh                   s
hadoop-daemons.sh      mr-jobhistory-daemon.sh  s
hdfs-config.cmd        refresh-namenodes.sh     s
hdfs-config.sh         slaves.sh                s
hduser@laptop:/usr/local/hadoop/sbin$
hduser@laptop:/usr/local/hadoop/sbin$ stop-all.sh
This script is Deprecated. Instead use stop-dfs.s
15/04/18 15:46:31 WARN util.NativeCodeLoader: Una
Stopping namenodes on [localhost]
localhost: stopping namenode
localhost: stopping datanode
Stopping secondary namenodes [0.0.0.0]
0.0.0.0: no secondarynamenode to stop
15/04/18 15:46:59 WARN util.NativeCodeLoader: Una
stopping yarn daemons
stopping resourcemanager
localhost: stopping nodemanager
no proxyserver to stop
```

Hadoop Web Interfaces

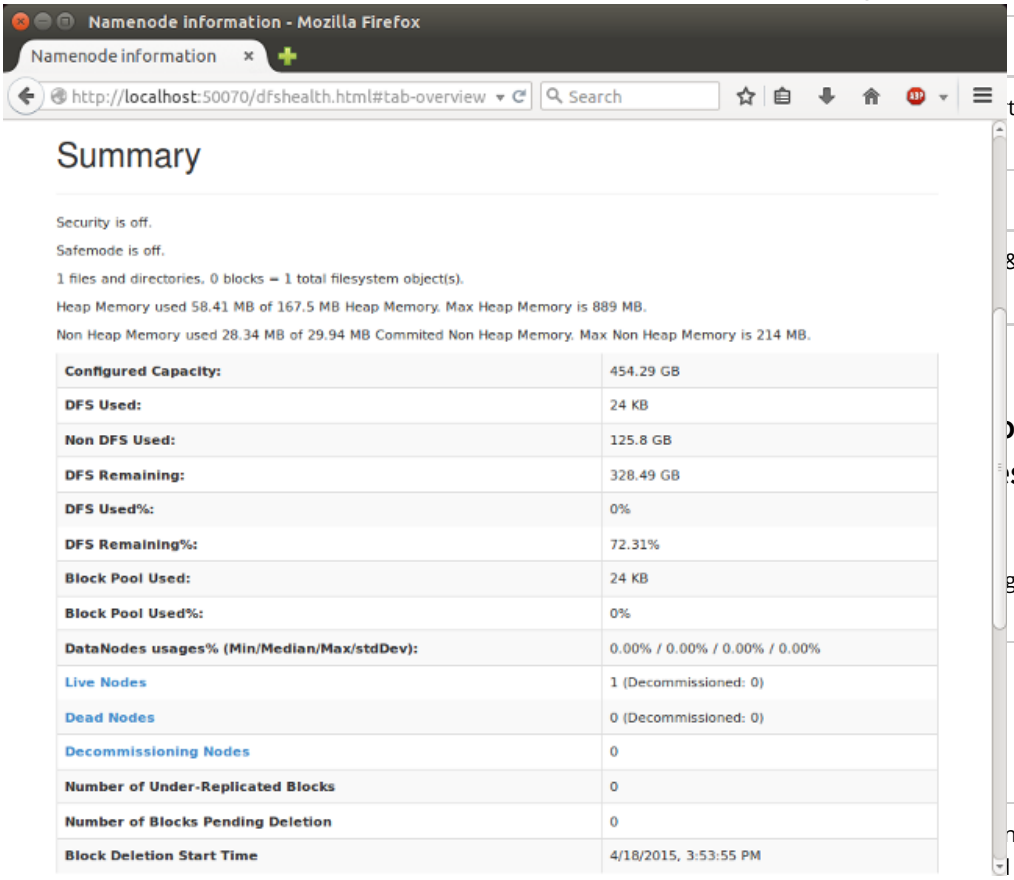
Let's start the Hadoop again and see its Web UI:

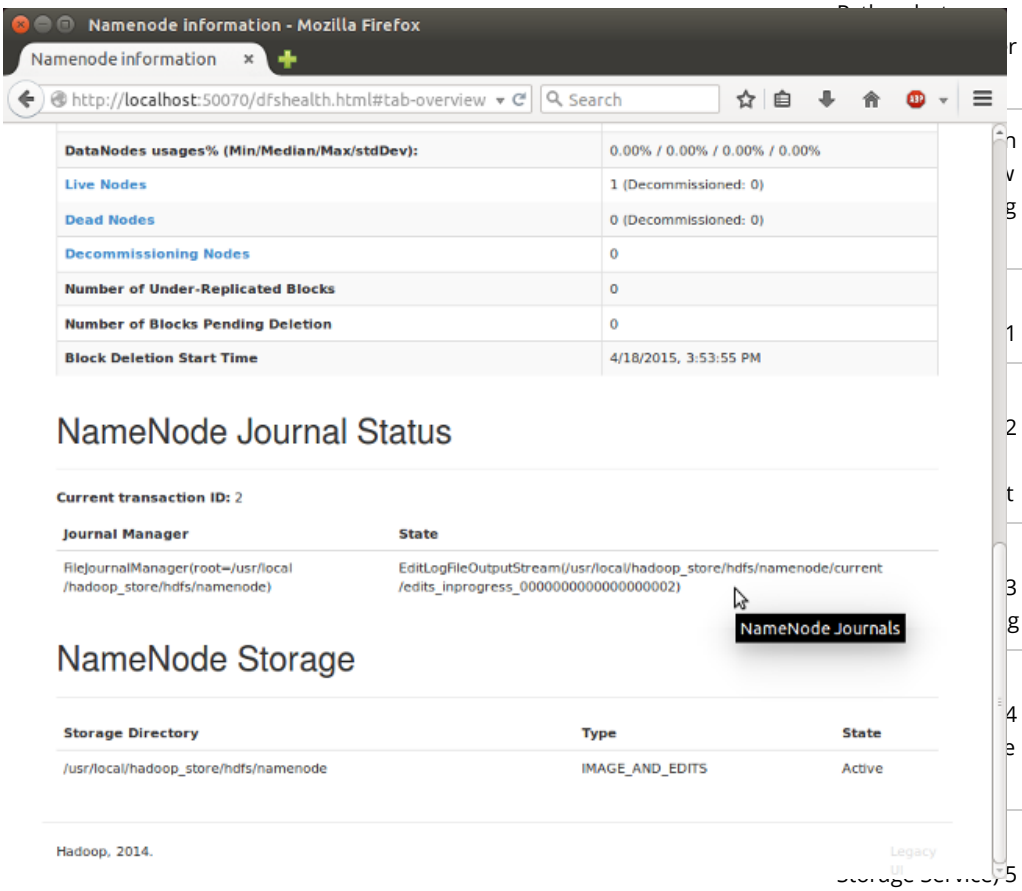
```
hduser@laptop:/usr/local/hadoop/sbin$ start-all.s
```

http://localhost:50070/ - web UI of the NameNode daemon



Creating a VirtualBox using Vagrant





SecondaryNameNode

attaching Amazon EBS volume to the instance using

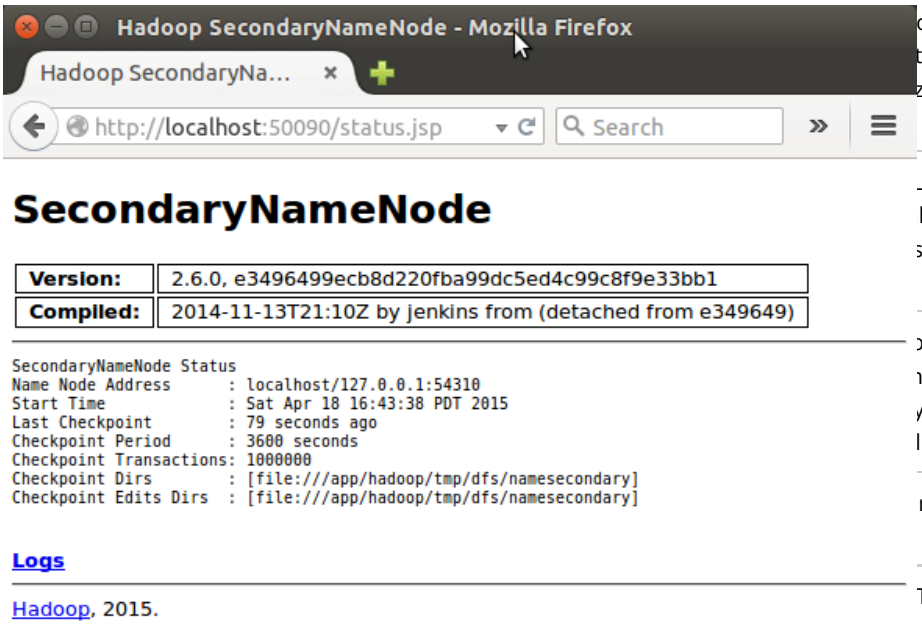
Uploading folders/files recursively

AWS : S3 (Simple Storage Service) 6 - Bucket Policy for File/Folder View/Download

AWS : S3 (Simple Storage Service) 7 - How to Copy or Move Objects from one region to another

AWS : S3 (Simple Storage Service) 8 - Archiving S3 Data to Glacier

AWS : Creating a



(Note) I had to restart Hadoop to get this Secondary Namenode.

DataNode

ont
tion with
zon S3

.B (Elastic
lancer) :
s, health

oad
g with
y (High
lity Proxy)

rtualBox

TP setup

AWS & OpenSSL : Creating / Installing a Server SSL Certificate

AWS : OpenVPN Access Server 2 Install

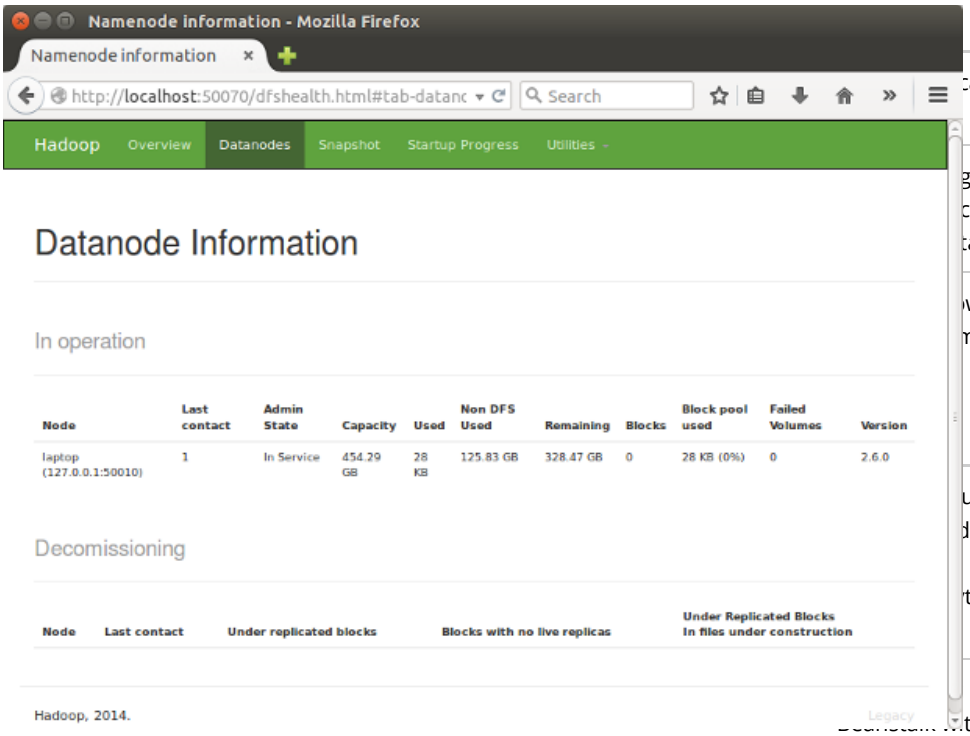
AWS : VPC (Virtual Private Cloud) 1 - netmask, subnets, default gateway, and CIDR

AWS : VPC (Virtual Private Cloud) 2 - VPC Wizard

AWS : VPC (Virtual Private Cloud) 3 - VPC Wizard with NAT

DevOps / Sys Admin Q & A (VI) - AWS VPC setup (public/private subnets with NAT)

AWS - OpenVPN Protocols : PPTP, L2TP/IPsec, and



caling

g a
count
tance

ows
mote

using

uled
d

thon

Legacy

NodeJS

AWS : Detecting stopped instance and sending an alert email using Mandrill smtp

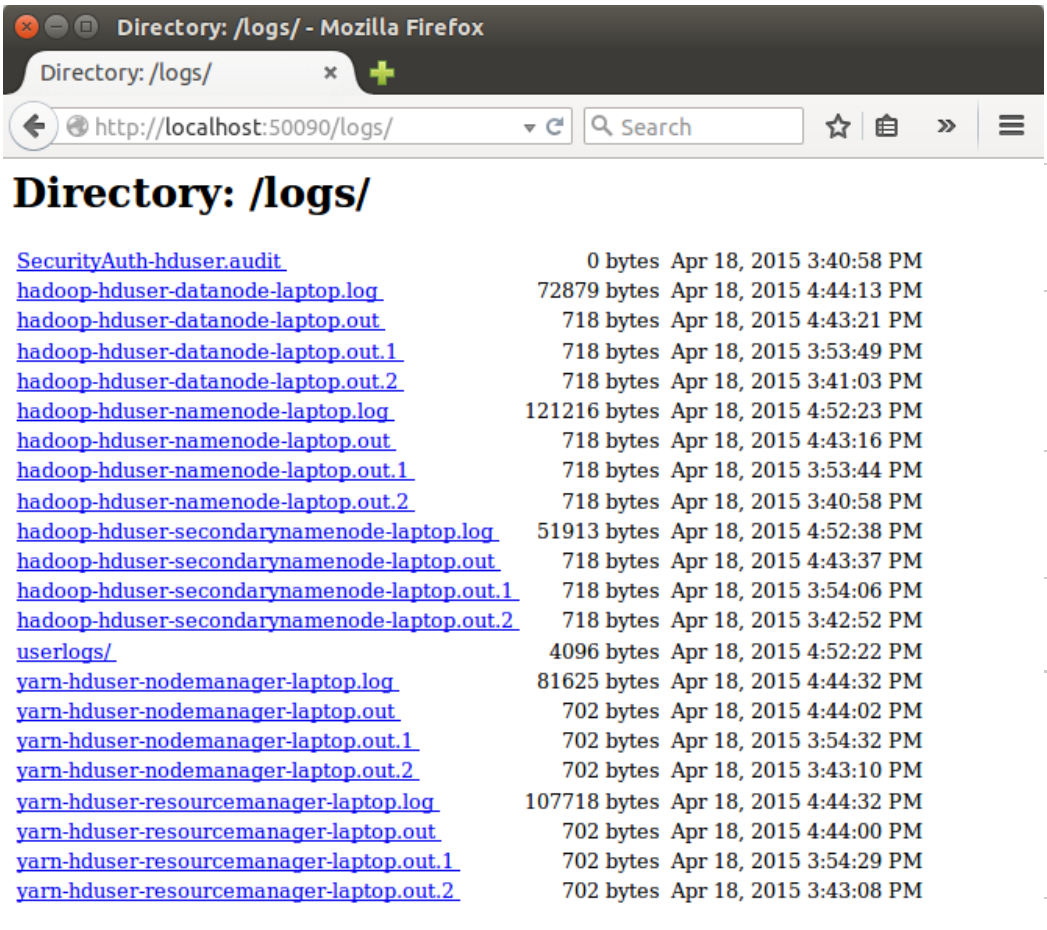
AWS : Identity and Access Management (IAM) Roles for Amazon EC2

AWS : Amazon Route 53

AWS : Amazon Route 53 - DNS (Domain Name Server) setup

AWS : Redshift data warehouse

AWS : RDS Connecting to a DB Instance Running the SQL Server Database Engine



Postgres on EC2 instance from S3 backup

Bogotobogo's contents

To see more items, click left or right arrow.

I hope this site is informative and helpful.

Redis

Redis vs Memcached

Redis 3.0.1 Install

rshell 4 tutorial

Powersehll : Introduction

Powersehll : Help System

Using Hadoop

If we have an application that is set up to use Hadoop, we can fire that up and start using it with our Hadoop installation!

Big Data & Hadoop Tutorials

- Hadoop 2.6 - Installing on Ubuntu 14.04 (Single-Node Cluster)
- Hadoop - Running MapReduce Job
- Hadoop - Ecosystem
- CDH5.3 Install on four EC2 instances (1 Name node and 3 Datanodes) using Cloudera Manager 5
- CDH5 APIs
- QuickStart VMs for CDH 5.3
- QuickStart VMs for CDH 5.3 II - Testing with wordcount
- QuickStart VMs for CDH 5.3 II - Hive DB query
- Scheduled start and stop CDH services
- Zookeeper & Kafka Install
- Zookeeper & Kafka - single node single broker cluster
- Zookeeper & Kafka - single node multiple broker cluster
- OLTP vs OLAP
- Apache Hadoop Tutorial I with CDH - Overview
- Apache Hadoop Tutorial II with CDH - MapReduce Word Count
- Apache Hadoop Tutorial III with CDH - MapReduce Word Count 2
- Apache Hadoop (CDH 5) Hive Introduction
- CDH5 - Hive Upgrade to 1.3 to from 1.2
- Apache Hadoop : Creating HBase table with HBase shell and HUE
- Apache Hadoop : Creating HBase table with Java API

Git/GitHub Tutorial

- Apache Hadoop : HBase in Pseudo-Distributed mode
- Apache Hadoop HBase : Map, Persistent, Sparse, Sorted, Distributed and Multidimensional
- Apache Hadoop - Flume with CDH5: a single-node Flume deployment (telnet example)
- Apache Hadoop (CDH 5) Flume with VirtualBox : syslog example via NettyAvroRpcClient
- List of Apache Hadoop hdfs commands
- Apache Hadoop : Creating Wordcount Java Project with Eclipse Part 1
- Apache Hadoop : Creating Wordcount Java Project with Eclipse Part 2
- Apache Hadoop : Creating Card Java Project with Eclipse using Cloudera VM UnoExample for CDH5 - local run
- Apache Hadoop : Creating Wordcount Maven Project with Eclipse
- Wordcount MapReduce with Oozie workflow with Hue browser - CDH 5.3 Hadoop cluster using VirtualBox and QuickStart VM
- Spark 1.2 using VirtualBox and QuickStart VM - wordcount
- Spark Programming Model : Resilient Distributed Dataset (RDD)
- Apache Spark 1.3 with PySpark (Spark Python API) Shell
- Apache Spark 1.2 with PySpark (Spark Python API) Wordcount using CDH5
- Apache Spark 1.2 Streaming

Subversion