1 Εγκατάσταση Hadoop 2.6.3 cluster

Στο κείμενο αυτό περιγράφεται η εγκατάσταση του Hadoop framework στην υποδομή του okeanos ή σε μια υποδομή virtualization όπως το virtual box. Περιγράφεται επίσης πως μπορούμε να επεξεργαστούμε κώδικα μέσω του eclipse IDE και εν συνεχεία να τον εκτελέσουμε στο Hadoop cluster που έχουμε δημιουργήσει. Ο οδηγός αυτός είναι βασισμένος στις οδηγίες που δίνονται στην επίσημη σελίδα του Hadoop(http://hadoop.apache.org/docs/r2.6.3/) και μπορεί να χρησιμοποιηθεί για οποιοδήποτε Debian-based Linux σύστημα.

Για να γίνει η εγκατάσταση του Hadoop cluster στον okeanos, θα χρειαστεί να εργαστούμε σ' ένα απομακρυσμένο(remote) περιβάλλον. Για να το πετύχουμε αυτό, χρησιμοποιούμε τα προγράμματα ssh και scp αν δουλεύουμε σε Linux ή Mac και putty και winscp σε περίπτωση που δουλεύουμε σε Windows. Για τη χρήση αυτών των προγραμμάτων συμβουλευόμαστε το Google και τα manpages του λειτουργικού συστήματος(π.χ. man ssh).

Η εγκατάσταση περιλαμβάνει τα εξής συστήματα:

- HDFS: Κατανεμημένο filesystem, από το οποίο διαβάζουν και γράφουν τα Hadoop jobs.
- Hadoop: Open source υλοποίηση του προγραμματιστικού μοντέλου MapReduce
- YARN: Σύστημα που αναλαμβάνει το scheduling και monitoring των εργασιών στα διαθέσιμα resources του cluster.

Οι υπολογιστές(κόμβοι) ενός Hadoop cluster μπορεί να έχουν έναν ή παραπάνω από τους παρακάτω ρόλους:

HDFS DataNode:	Οι DataNodes περιέχουν κομμάτια(blocks) από τα αρχεία του HDFS. Αναλαμβάνουν να «σερβίρουν» δεδομένα σε κλήσεις εξωτερικών πελατών.
HDFS NameNode:	Πρόκειται για τον κεντρικό κόμβο του HDFS. Ο NameNode περιέχει την πληροφορία με την αντιστοίχιση των blocks των αρχείων με τους αντίστοιχους DataNodes (δηλ σε ποιον DataNode βρίσκεται κάθε block)
YARN ResourceManager:	Ο ResourceManager (RM) είναι ο κεντρικός κόμβος (master) του YARN που ελέγχει τους διαθέσιμους πόρους και με βάση αυτούς δρομολογεί και διαχειρίζεται τις κατανεμημένες εφαρμογές που τρέχουν στο cluster(π.χ. Hadoop).
YARN NodeManager:	Ο NodeManager (NM) είναι μια διεργασία του YARN, η οποία τρέχει σε κάθε κόμβο του cluster και διαχειρίζεται τους workers

	των κατανεμημένων εφαρμογών που τρέχουν στον κόμβο αυτό.

1.1 Δημιουργία μηχανημάτων

1.1.1 Μέσω του okeanos

Από το site του okeanos (https://cyclades.okeanos.grnet.gr/ui/) δημιουργούμε 2 εικονικές μηχανές (virtual machines) τις οποίες ονομάζουμε hadoop master και hadoop slave. Ο hadoop master θα παίζει τον ρόλο του HDFS NameNode και YARN ResourceManager. Αντίστοιχα, ο slave θα παίζει τον ρόλο του DataNode και NodeManager. Για το λειτουργικό σύστημα επιλέγουμε Debian, και για το hardware των virtual machines επιλέγουμε 1 πυρήνα, 2GB RAM και 10GB σκληρό δίσκο. Μετά από την επιτυχημένη δημιουργία των μηχανημάτων ο okeanos επίστρέφει το IPv6 των μηχανημάτων μαζί με το root password για να μπορέσουμε να συνδεθούμε την πρώτη φορά. Φροντίζουμε να κρατήσουμε τους κωδικούς του root χρήστη για τα μηχανήματα.

TIP: Επειδή καμιά φορά λόγω κακού connectivity, μπορεί να χάσουμε το δίκτυο και να μην προλάβουμε να πάρουμε τον κωδικό, καλό είναι από το tab με το κλειδί να φτιάξουμε ένα key-pair και να το προσθέσουμε κατά την εγκατάσταση.

Στη συνέχεια πηγαίνουμε στο tab του site του okeanos με το όνομα «IP» και δημιουργούμε 1 καινούρια IPv4 διευθύνση και την κάνουμε attach στο hadoop master μηχάνημα που δημιουργήσαμε. Μετά πηγαίνουμε στο tab με τα networks και δημιουργούμε ένα καινούριο ιδιωτικό δίκτυο. Μόλις δημιουργηθεί, από το UI του okeanos, συνδέουμε τα VMs μας στο δίκτυο αυτό. Με αυτό τον τρόπο καταφέραμε να έχουν και τα δύο εικονικά μηχανήματα private IPv4. Τις IPv4 τις χρειαζόμαστε γιατί είναι απαραίτητες για τη λειτουργία του Hadoop.

1.1.2 Μέσω virtual box

Χρειάζεται να έχουμε το virtualbox που τρέχει εικονικές μηχανές τόσο από Windows όσο και από Linux μηχανήματα. Κατεβάζουμε την σωστή έκδοση για το λειτουργικό μας από εδώ: https://www.virtualbox.org/wiki/Downloads

Από τον wizard επιλέγουμε New machine->Linux-Debian. Του δίνουμε 384MB RAM (default) και στην επόμενη οθόνη επιλέγουμε Startup Disk και create new hard disk (8GB), τύπο VDI, και dynamically allocated χώρο. Η παρεχόμενη RAM και ο ελεύθερος χώρος μπορούν αργότερα να αλλάξουν εάν ο χρήστης το επιθυμεί.

Κατόπιν κατεβάζουμε το cd εγκατάστασης netinst iso του Debian (~190MB) .

```
wget http://cdimage.debian.org/debian-cd/6.0.4/i386/iso-cd/debian-6.0.4-i386-businesscard.iso
```

Κατά την πρώτη εκκίνηση του virtual machine θα ζητηθεί ένα bootable image για να ξεκινήσει το λειτουργικό. Του δίνουμε το iso που κατεβάσαμε από το debian site, και συνεχίζουμε την εγκατάσταση. Στο τέλος θα έχουμε ένα virtual machine με παρόμοιο software με αυτό του okeanos. Το καλό με αυτό το virtual machine είναι ότι μπορούμε να το μεταφέρουμε σε οποιοδήποτε σύστημα τρέχει virtual box (ανεξάρτητα εάν είναι Linux ή Windows).

1.2 Προαιρετικό: ενημέρωση του αρχείου hosts του μηχανήματός μας.

Με αυτό το βήμα δεν χρειάζεται να θυμόμαστε απ'έξω τα IP addresses των μηχανημάτων του cluster που στήσαμε στον okeanos.

Το αρχείο hosts υπάρχει σε κάθε υπολογιστή, και στην ουσία αποτελεί μια τοπική βάση DNS την οποία συμβουλεύεται πρώτη κάθε φορά που προσπαθεί να κάνει resolve ένα όνομα σε διεύθυνση IP. Το αρχείο hosts περιέχει αντιστοιχίσεις ονομάτων σε IPs. Το αρχείο αυτό σε συστήματα Windows βρίσκεται στην τοποθεσία

```
C:\Windows\System32\drivers\etc\hosts
```

Ενώ σε συστήματα Linux είναι στην τοποθεσία

```
/etc/hosts
```

Ανοίγουμε το αρχείο hosts και συμπληρώνουμε στο τέλος:

```
<okeanos_master_ip>     master
<okeanos_slave_ip>      slave
```

Από εδώ και πέρα, ο υπολογιστής μας κάνει resolve τα DNS names master και slave στα IPs των μηχανημάτων του okeanos. Έτσι, δεν χρειάζεται να θυμόμαστε απ'έξω το IP του κάθε μηχανήματος.

Συνδεόμαστε με ssh στον hadoop master (δίνουμε σαν κωδικό τον κωδικό που μας έδωσε ο okeanos ή αυτόν που επιλέξαμε κατά την εγκατάσταση του virtual machine).

```
ssh root@master
```

τρέχουμε

```
echo 'master' > /etc/hostname
```

για να ονομάσουμε το μηχάνημα master. Κάνουμε το ίδιο και για το μηχάνημα slave.

```
ssh root@slave
```

τρέχουμε

```
echo 'slave' > /etc/hostname
```

1.3 Ενημέρωση αρχείου hosts των hadoop machines

Στο μηχάνημα master συνδεόμαστε σαν root και τρέχουμε

```
ssh root@master
vi /etc/hosts
```

και στο τέλος του αρχείου βάζουμε

```
<okeanos_master_ip>     master
<okeanos_slave_ip>      slave
```

Κάνουμε το ίδιο και για το μηχάνημα slave:

```
ssh root@slave
vi /etc/hosts
```

και στο τέλος του αρχείου βάζουμε

```
<okeanos_master_ip>     master
<okeanos_slave_ip>      slave
```

1.4 Ρύθμιση για passwordless ssh

Το Hadoop για να λειτουργήσει θα πρέπει οι υπολογιστές που αποτελούν το cluster να μπορούν να συνδέονται μεταξύ τους με την χρήση ssh χωρίς κωδικό. Αυτό επιτυγχάνεται με την χρήση ζεύγους ιδιωτικού/δημόσιου κλειδιού[1]. Η βασική ιδέα της τεχνικής αυτής είναι ότι ο κάθε χρήστης μπορεί να έχει ένα ιδιωτικό κλειδί (αρχείο) το οποίο έρχεται ζεύγος με ένα δημόσιο κλειδί. Ο χρήστης τοποθετεί το δημόσιο κλειδί στους υπολογιστές τους οποίους θέλει να έχει πρόσβαση, και με την χρήση του ιδιωτικού μπορεί να συνδέεται σε αυτούς χωρίς κωδικό. Η πιστοποίηση του χρήστη γίνεται καθώς μόνο ο χρήστης έχει στην κατοχή του το ιδιωτικό κλειδί.

```
ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
```

η εντολή δημιουργεί ένα ιδιωτικό κλειδί id_rsa και ένα δημόσιο κλειδί id_rsa.pub. Τρέχουμε

```
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

η παραπάνω εντολή βάζει το δημόσιο κλειδί id_rsa.pub στον κατάλογο με τα αποδεκτά δημόσια κλειδιά του χρήστη root του master μηχανήματος. Με αυτόν τον τρόπο, όποιος έχει στην κατοχή του το ιδιωτικό κλειδί id_rsa μπορεί να συνδεθεί στο root@master.

Τώρα, αρκεί να μεταφέρουμε τον κατάλογο .ssh στον χρήστη root και του μηχανήματος slave

```
scp -r .ssh/ root@slave:~/
```

1.5 Δημιουργία NAT

Από τα δύο VMs που φτιάξαμε και τα δύο έχουν private IPv4, αλλά μόνο ο master έχει public IPv4. Αυτό σημαίνει ότι στο ισχύον setup μόνο αυτός μπορεί να βλέπει μέσω δικτύου στον «έξω κόσμο». Επειδή θα χρειαστεί να εγκαταστήσουμε διάφορα πακέτα και στον slave, θέλουμε να δώσουμε και σε αυτόν πρόσβαση προς τα έξω.

Για να το πετύχουμε αυτό τρέχουμε στον master το παρακάτω script:

```
#!/bin/bash
echo "Enabling ipv4 forwarding (cleaning old rules)"
# flushing old rules -- USE WITH CARE
```

---

[1] http://en.wikipedia.org/wiki/Public-key_cryptography

```
iptables --flush
iptables --table nat --flush
# MASQUERADE each request form the inside to the outer world
iptables -t nat -A POSTROUTING -j MASQUERADE
# enable IPv4 packet forwarding in the kernel
echo 1 > /proc/sys/net/ipv4/ip_forward
echo "Master is not operating as router"
```

και στον slave:

```
#!/bin/bash
ENDPOINT_INTERFACE=$(cat /etc/hosts | grep master | awk '{print $1}')
route add default gw $ENDPOINT_INTERFACE
echo "Gateway now points to $ENDPOINT_INTERFACE"
```

1.6 Βελτίωση Shell

Εγκαθιστούμε το vim αντί για vi.

```
apt-get install vim
```

Ενεργοποιούμε το syntax highlight στο vim.

```
vim /etc/vim/vimrc
```

Στο τέλος του αρχείου προσθέτουμε την εντολή «syntax on».

Κάνουμε edit το .bashrc που βρίσκεται στο home dir μας και ενεργοποιούμε colors κατά την εκτέλεση του ls.

```
vim ~/.bashrc
```

Κάνουμε uncomment τα “`alias ls='ls --color=auto'`”

Ενεργοποιούμε το bash autocompletion feature για να παίζει το διπλό tab

```
apt-get install bash-completion
```

1.7 Εγκατάσταση oracle java-7

Για να εγκαταστήσουμε την oracle java 7 πρέπει πρώτα να ενημερώσουμε το source.list του apt ώστε να βρει το λειτουργικό τα κατάλληλα repositories πακέτων.

```
apt-get install software-properties-common
echo "deb http://ppa.launchpad.net/webupd8team/java/ubuntu precise
main" | tee /etc/apt/sources.list.d/webupd8team-java.list

echo "deb-src http://ppa.launchpad.net/webupd8team/java/ubuntu
precise main" | tee -a /etc/apt/sources.list.d/webupd8team-java.list

apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys
EEA14886

apt-get update
apt-get install oracle-java7-installer
```

Μόλις ενημερώσουμε τα repositories κι εγκαταστήσουμε την java, ελέγχουμε αν η εγκατάσταση ολοκληρώθηκε με επιτυχία.

```
root@master:~# java –version
java version "1.7.0_80"
Java(TM) SE Runtime Environment (build 1.7.0_80-b15)
Java HotSpot(TM) 64-Bit Server VM (build 24.80-b11, mixed mode)
```

Η παραπάνω διαδικασία πρέπει να ακολουθηθεί και για το slave μηχάνημα.

1.8 Εγκατάσταση hadoop

Τα παρακάτω βήματα πρέπει να γίνουν σε όλους τους υπολογιστές του cluster.

Κατεβάζουμε το installation package από ένα repository και το αποσυμπιέζουμε στον κατάλογο /opt.

```
cd /opt
wget http://www.eu.apache.org/dist/hadoop/common/hadoop-2.6.3/hadoop-2.6.3.tar.gz
tar -xzf hadoop-2.6.3.tar.gz
```

Κάνουμε edit to ~/.bashrc ώστε να κάνουμε export τις παρακάτω μεταβλητές περιβάλλοντος. Προσθέτουμε τις παρακάτω γραμμές στο τέλος του bashrc.

```
export JAVA_HOME=/usr/lib/jvm/java-7-oracle/
export HADOOP_INSTALL=/opt/hadoop-2.6.3/
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_HOME=$HADOOP_INSTALL
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export HADOOP_YARN_HOME=$HADOOP_INSTALL
export HADOOP_CONF_DIR=$HADOOP_INSTALL/etc/hadoop
```

Το configuration του Hadoop βρίσκεται στο path: $HADOOP_INSTALL/etc/hadoop. Στον κατάλογο αυτό θα ρυθμίσουμε τα εξής αρχεία:

- core.xml
- hdfs-site.xml: Ρυθμίσεις για την παραμετροποίηση του HDFS
- yarn-site.xml: Ρυθμίσεις για την παραμετροποίηση του YARN
- mapred-site.xml: Ρυθμίσεις για την παραμετροποίηση του περιβάλλοντος εκτέλεσης ενός MapReduce job
- slaves: τα host names των κόμβων που λειτουργούν ως compute slaves

Περισσότερες λεπτομέρειες για τα αρχεία παραμετροποίησης μπορούν να βρεθούν στη διεύθυνση: http://hadoop.apache.org/docs/r2.6.3/hadoop-project-dist/hadoop-common/ClusterSetup.html

Κάνουμε edit τα αρχεία (vim <file-name>) και προσθέτουμε τα εξής περιεχόμενα:

core-site.xml

```
<?xml version="1.0"?>
```

```
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Put site-specific property overrides in this file. -->
<configuration>
      <property>
            <name>fs.default.name</name>
            <value>hdfs://master:9000</value>
      </property>
</configuration>
```

hdfs-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Put site-specific property overrides in this file. -->
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
    <description>Default block replication.</description>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/opt/hdfsnames</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/opt/hdfsdata</value>
  </property>
  <property>
    <name>dfs.blocksize</name>
    <value>64m</value>
    <description>Block size</description>
  </property>
  <property>
    <name>dfs.webhdfs.enabled</name>
    <value>true</value>
  </property>
  <property>
    <name>dfs.support.append</name>
    <value>true</value>
  </property>
</configuration>
```

mapred-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Put site-specific property overrides in this file. -->
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
    <description>No description</description>
  </property>
  <property>
    <name>mapreduce.map.java.opts</name>
    <value>-Xmx1500m</value>
    <description>No description</description>
  </property>
  <property>
  <name>mapreduce.map.memory.mb</name>
```

```
    <value>1800</value>
    <description>No description</description>
  </property>
  <property>
    <name>mapreduce.reduce.java.opts</name>
    <value>-Xmx1500m</value>
    <description>No description</description>
  </property>
  <property>
  <name>mapreduce.reduce.memory.mb</name>
    <value>1800</value>
    <description>No description</description>
  </property>
  <property>
    <name>mapreduce.map.cpu.vcores</name>
    <value>1</value>
    <description>No description</description>
  </property>
  <property>
    <name>mapreduce.task.timeout</name>
    <value>0</value>
  </property>
</configuration>
```

yarn-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<!-- Site specific YARN configuration properties -->
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>master</value>
    <final>true</final>
    <description>host is the hostname of the resource manager.
    </description>
  </property>
  <property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>master:8025</value>
    <final>true</final>
    <description>host is the hostname of the resource manager and
    port is the port on which the NodeManagers contact the Resource
Manager.
    </description>
  </property>
  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>master:8030</value>
    <final>true</final>
    <description>host is the hostname of the resourcemanager and port
is the port
    on which the Applications in the cluster talk to the Resource
Manager.
    </description>
  </property>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>master:8050</value>
    <final>true</final>
    <description>the host is the hostname of the ResourceManager and
```

```
the port is the port on
    which the clients can talk to the Resource Manager.
</description>
  </property>
<property>
    <name>yarn.nodemanager.address</name>
    <value>0.0.0.0:0</value>
    <description>the nodemanagers bind to this port</description>
  </property>
  <property>
    <name>yarn.nodemanager.remote-app-log-dir</name>
    <value>/app-logs</value>
    <description>directory on hdfs where the application logs are
moved to </description>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
    <description>Shuffle service that needs to be set for Map Reduce
to run </description>
  </property>
  <property>
    <name>yarn.nodemanager.aux-
services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address</name>
    <value>master:8088</value>
    <description>The http address of the RM web application
</description>
  </property>
  <property>
    <name>yarn.nodemanager.vmem-check-enabled</name>
    <value>false</value>
  </property>
  <property>
    <name>yarn.nodemanager.resource.memory-mb</name>
    <value>1800</value>
  </property>
  <property>
    <name>yarn.scheduler.minimum-allocation-mb</name>
    <value>1800</value>
  </property>
  <property>
    <name>yarn.scheduler.maximum-allocation-mb</name>
    <value>1800</value>
  </property>
  <property>
    <name>yarn.nodemanager.resource.cpu-vcores</name>
    <value>1</value>
  </property>
</configuration>
```

1.9 Εκκίνηση του HDFS

Το πρώτο πράγμα που έχουμε να κάνουμε είναι format του HDFS. Συνδεόμαστε στον master και δίνουμε την παρακάτω εντολή:

```
ssh root@master
```

```
hdfs namenode -format
```

Format κάνουμε την πρώτη φορά που εγκαθιστούμε το Hadoop.

Στη συνέχεια ξεκινάμε το HDFS(NameNode + DataNodes) δίνοντας από τον master την εξής εντολή:

```
start-dfs.sh
```

Παίρνουμε το παρακάτω output:

```
root@master:~# start-dfs.sh
Starting namenodes on [master]
The authenticity of host 'master (83.212.104.253)' can't be
established.
ECDSA key fingerprint is
03:f4:58:65:d0:ca:76:b6:53:50:bf:1f:cd:d9:2a:2b.
Are you sure you want to continue connecting (yes/no)? yes
master: Warning: Permanently added 'master,83.212.104.253' (ECDSA) to
the list of known hosts.
master: starting namenode, logging to /opt/hadoop-2.6.2/logs/hadoop-
root-namenode-master.out
slave: starting datanode, logging to /opt/hadoop-2.6.2/logs/hadoop-
root-datanode-slave.out
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
ECDSA key fingerprint is
03:f4:58:65:d0:ca:76:b6:53:50:bf:1f:cd:d9:2a:2b.
Are you sure you want to continue connecting (yes/no)? yes
0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of
known hosts.
0.0.0.0: starting secondarynamenode, logging to /opt/hadoop-
2.6.2/logs/hadoop-root-secondarynamenode-master.out
```

Την πρώτη φορά πρέπει να πατήσουμε “y” κατά την σύνδεση του master με τον slave, καθώς ο master ξεκινάει τον slave μέσω ssh και δεν τον γνωρίζει.

Ελέγχουμε να δούμε αν τρέχει το HDFS:

```
root@master:~# jps
20024 NameNode
20310 Jps
20187 SecondaryNameNode
```

1.10 Εκκίνηση YARN(Hadoop)

Με παρόμοια διαδικασία ξεκινάμε το YARN, τρέχοντας στον master την εντολή:

```
start-yarn.sh
```

Παίρνουμε το παρακάτω output:

```
root@master:~# start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /opt/hadoop-2.6.2//logs/yarn-
root-resourcemanager-master.out
slave: starting nodemanager, logging to /opt/hadoop-2.6.2//logs/yarn-
```

```
root-nodemanager-slave.out
```

Πλέον βλέπουμε ότι στον master τρέχει και ο ResourceManager του YARN.

```
root@master:~# jps
20400 ResourceManager
20671 Jps
20024 NameNode
20187 SecondaryNameNode
```

Το σύστημα έχει ξεκινήσει και στα παρακάτω url μπορεί κανείς να βλέπει την κατάσταση του cluster:

http://master:50070 : Το url αυτό δείχνει την κατάσταση του NameNode, καθώς και τα περιεχόμενα του αποθηκευτικού συστήματος.

http://master:8088 : Αυτό είναι το url για το endpoint του YARN και δείχνει την κατάσταση των resources του cluster και των εφαρμογών που τρέχουν σε αυτό.

1.11 Χρήσιμες εντολές, αρχεία:

- Όπως είδαμε, με την εντολή jps μπορούμε να δούμε τις java διεργασίες που τρέχουν σ' ένα μηχάνημα. Κάνοντας kill -9 <pid> σκοτώνουμε κάποια διεργασία που δεν αποκρίνεται.

- Ο κατάλογος log/ περιέχει τα logfiles όλων των υπηρεσιών του Hadoop. Είναι χρήσιμος καθώς εκεί μπορούμε να βλέπουμε πιθανά λάθη/προβλήματα που προκύπτουν.

- Στον κατάλογο tmp υπάρχουν τα lock files των προγραμμάτων του Hadoop (με κατάληξη .pid). Καλό είναι, εάν έχουμε τερματίσει τον Hadoop cluster με βίαιο τρόπο (πχ με kill -9 <pid>), να τα σβήνουμε.

2 Εκτέλεση παραδειγμάτων

Για τα παραδείγματα θα χρειαστούμε κάποιο πρόγραμμα ανάπτυξης εφαρμογών java. Το eclipse είναι από τους πιο διαδεδομένους IDE για java (και όχι μόνο). Μπορούμε να κατεβάσουμε το eclipse IDE for java developers από τη διεύθυνση:

```
http://www.eclipse.org/downloads/
```

Στη συνέχεια κατεβάζουμε το source κώδικα του Hadoop 2.6.3 στον υπολογιστή μας. Το release με τον κώδικα μπορούμε να το κατεβάσουμε από το παρακάτω link: http://archive.apache.org/dist/hadoop/core/hadoop-2.6.3/

Πλέον είμαστε έτοιμοι να επεξεργαστούμε τον κώδικα των Hadoop examples. Για να τον εισάγουμε στο workspace του eclipse, πηγαίνουμε:

File > Import > Existing Maven Projects

Και αφού κάνουμε browse το filesystem μας στην τοποθεσία που κατεβάσαμε τον κώδικα του Hadoop, επιλέγουμε το project: hadoop-mapreduce-examples.

Αφότου εισαχθεί στο workspace, ανοίγουμε και κάνουμε inspect τον κώδικα του WordCount που βρίσκεται στο package org.apache.hadoop.examples. Για να τον κάνουμε compile, δίνουμε: Run as.. > Maven build και όταν μας ζητηθεί κάποιος goal, δίνουμε install φροντίζοντας να κάνουμε skip τα tests. Εναλλακτικά, μπορούμε να πάμε από το terminal στο path του filesystem, όπου βρίσκεται ο κώδικας με τα examples και να δώσουμε:

mvn install –Dmaven.test.skip=true

Μόλις ολοκληρωθεί το compile, στον φάκελο target έχει δημιουργηθεί ένα jar file με το όνομα: hadoop-mapreduce-examples-2.6.3.jar. Με χρήση του scp ή του WinScp (ανάλογα με το λειτουργικό μας σύστημα), μεταφέρουμε το αρχείο αυτό στον master του cluster που έχουμε στήσει στον okeanos.

Για να τρέξουμε το wordcount, δίνουμε την παρακάτω εντολή:

```
ssh root@master
yarn jar hadoop-mapreduce-examples-2.6.3.jar wordcount <input>
<output>
```