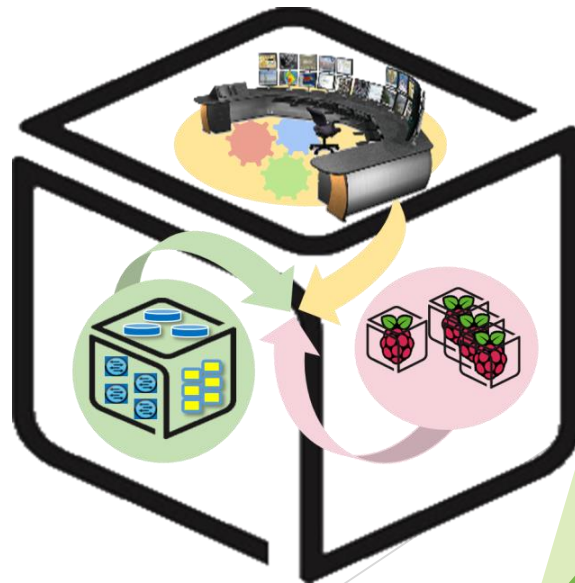


SmartX Labs for Computer Systems

Cluster Lab

(2016, Spring)

NetCS Lab



History and Contributor of Cluster Lab

(2016. 06. 01.)

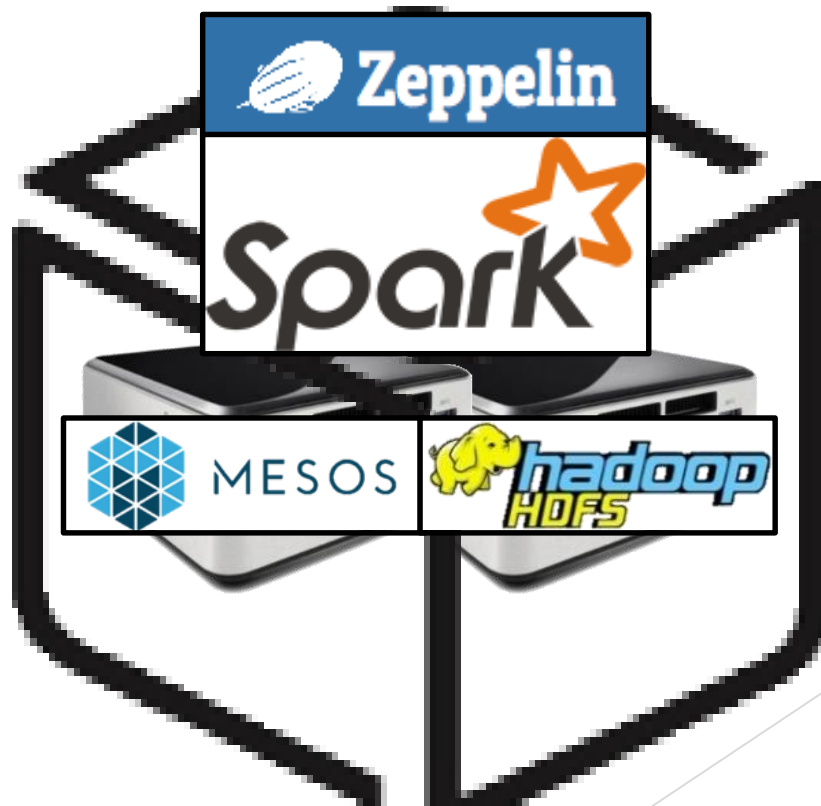
Version	Updated Date	Updated Contents	Contributor
-	2015/10	(구) Analytics Lab 작성	송지원
v1	2016/04	Cluster Lab 초안 작성	김승룡
v2	2016/05	Cluster Lab 수정	송지원
v3r3	2016/05/28	Cluster Lab 2차 수정 (내용 수정 및 추가)	송지원
v4r1	2016/05/30	Cluster Lab 3차 수정 (피드백 반영)	송지원
v5	2016/06/01	HDFS를 옵션으로 변경, 기타 문제 수정	송지원
v6r1	2016/06/03	실습자 검수 후 수정	송지원, 윤희범, 남택호

CSLab: Cluster LAB

- Goal

SETUP to run data processing and visualization

- Install Mesos, HDFS, Spark, Zeppelin on NUC



Apache Mesos

- Concept



What is Mesos?

A distributed systems kernel

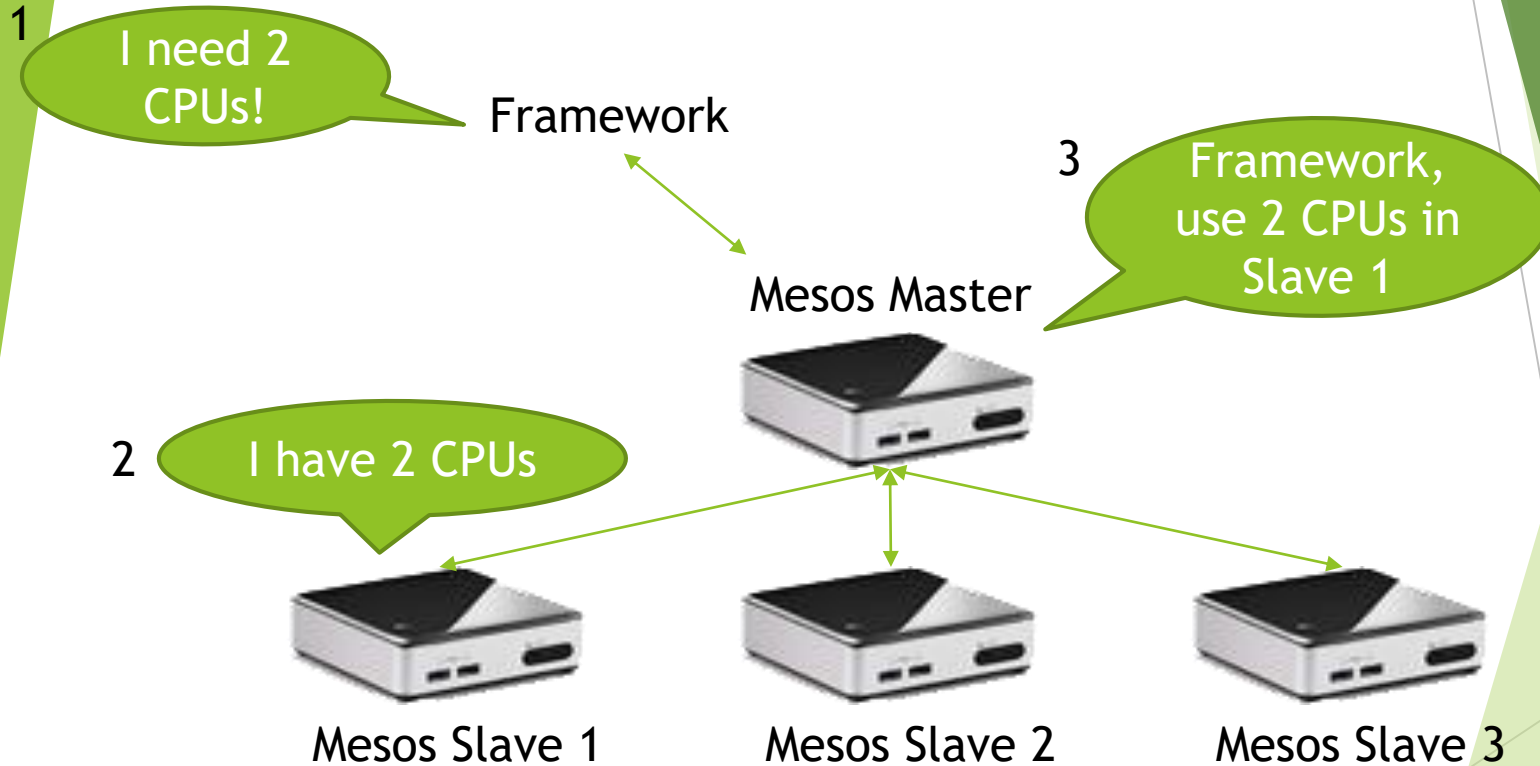
Mesos is built using the same principles as the Linux kernel, only at a different level of abstraction. The Mesos kernel runs on every machine and provides applications (e.g., Hadoop, Spark, Kafka, Elastic Search) with API's for resource management and scheduling across entire datacenter and cloud environments.

- Cloud as a single computer
- Share resources across the machines



Apache Mesos

- Architecture



HDFS

- Concept

Hadoop Distributed FileSystem

- A distributed file system that provides high-throughput access to application data.

Features

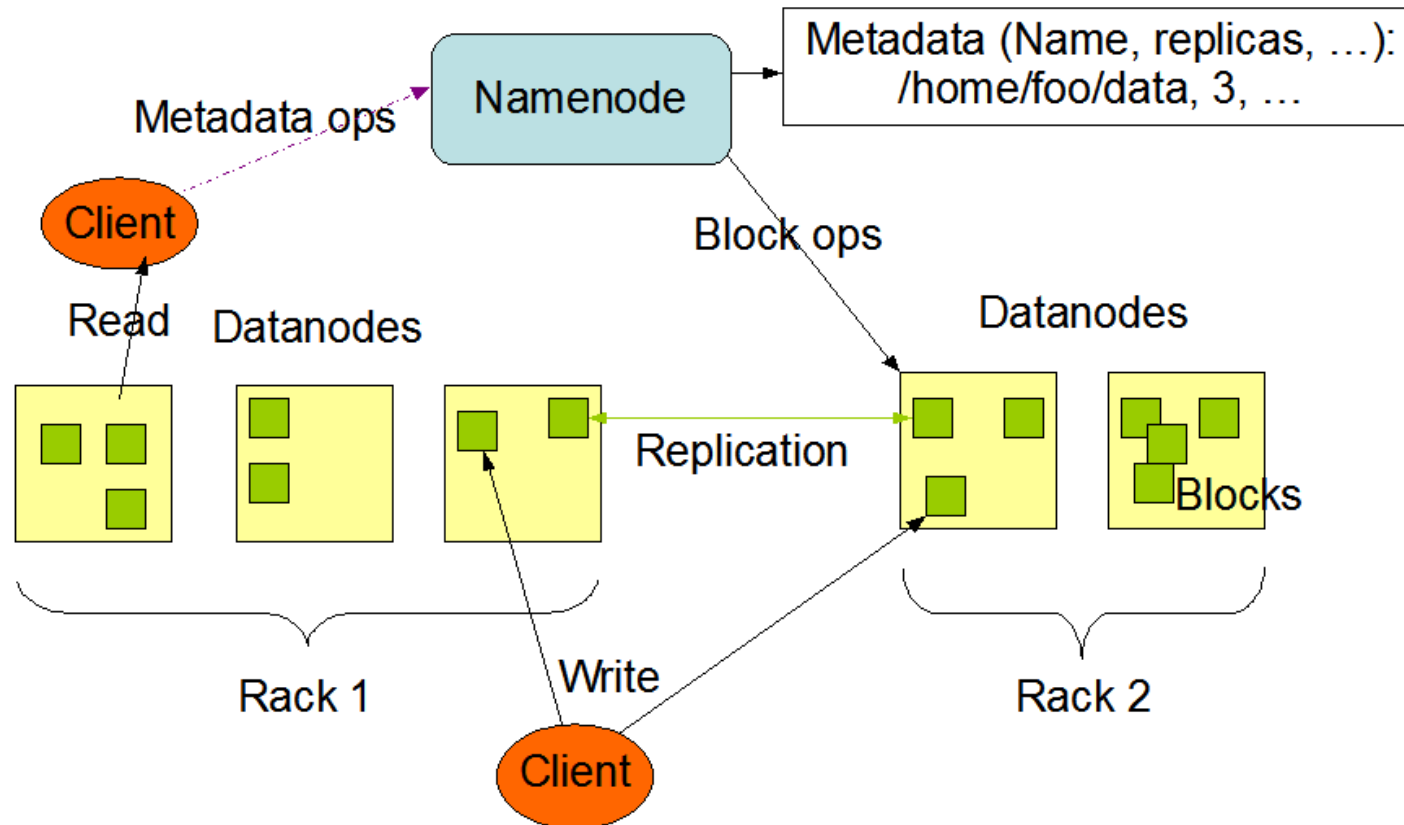
- **Fault tolerance** by detecting faults and applying quick, automatic recovery
- **Portability** across heterogeneous commodity hardware and operating systems
- **Scalability** to reliably store and process large amounts of data
- **Economy** by distributing data and processing across clusters of commodity personal computers
- **Efficiency** by distributing data and logic to process it in parallel on nodes where data is located
- **Reliability** by automatically maintaining multiple copies of data and automatically redeploying processing logic in the event of failures

HDFS

- Architecture

<Master/Slave architecture>

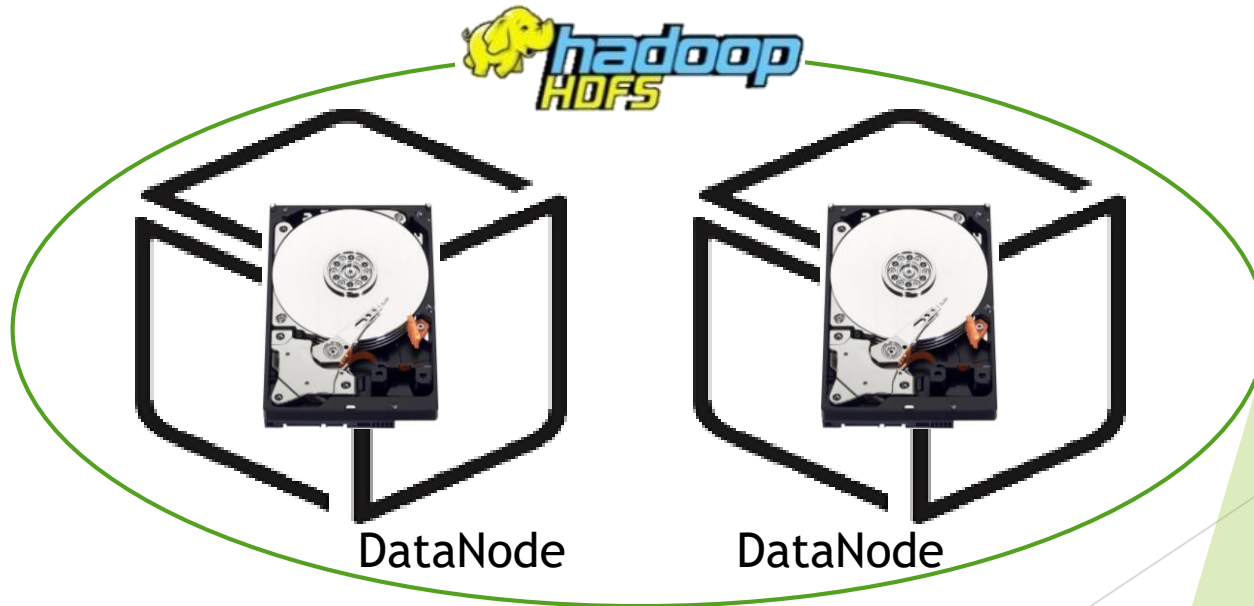
- NameNode: A single node which manages the file system namespace and regulates access to files by clients.
- DataNode: DataNodes manage storage attached to the nodes that they run on.



HDFS

- Architecture

HDFS makes storages of separate machines in cluster into a single storage.



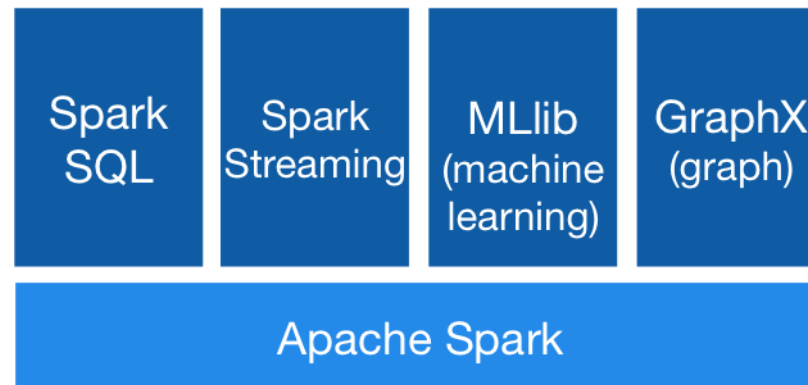
Apache Spark

- Concept



Apache Spark™ is a fast and general engine for large-scale data processing.

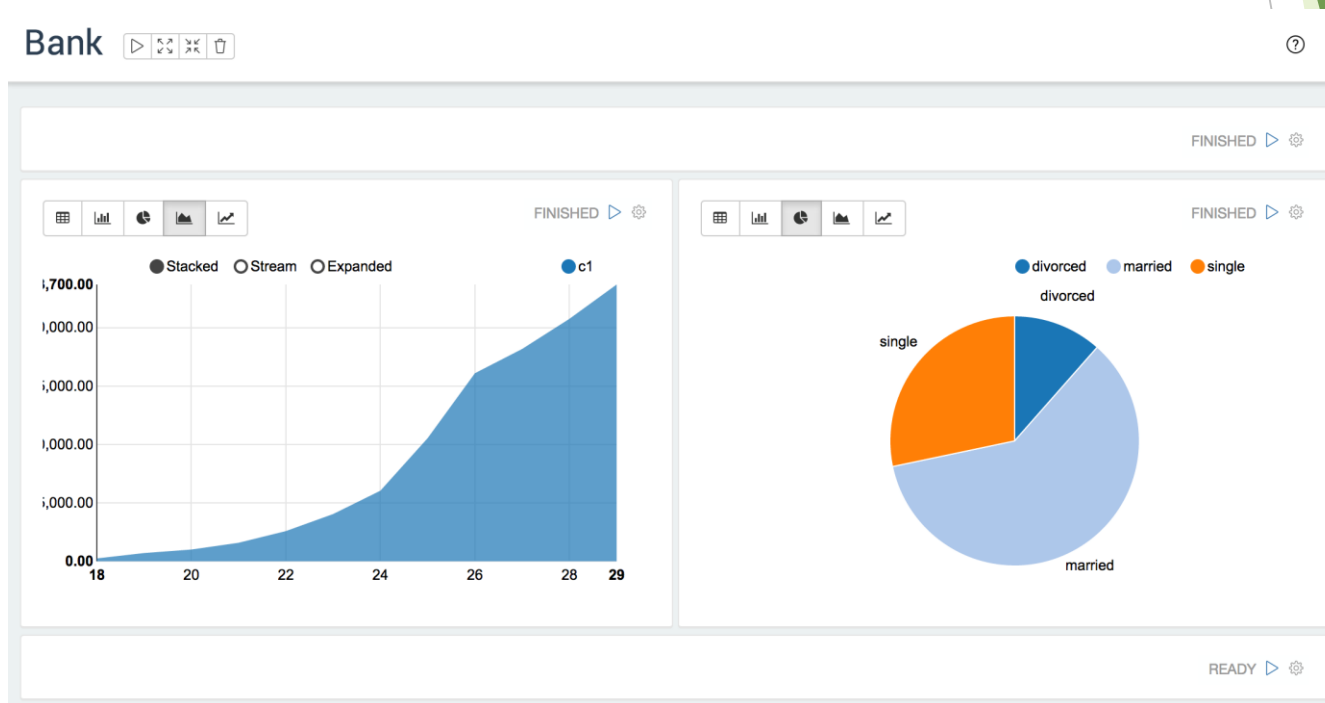
- In-memory data processing framework: Fast!
- Easy to use, community fastly growing
- Libraries: SQL and DataFrame, Streaming, MLlib, GraphX
- Run on standalone or Mesos, Yarn, etc
- Scala, Java, Python



Apache Zeppelin -Concept

A web-based notebook that enables interactive data analytics.

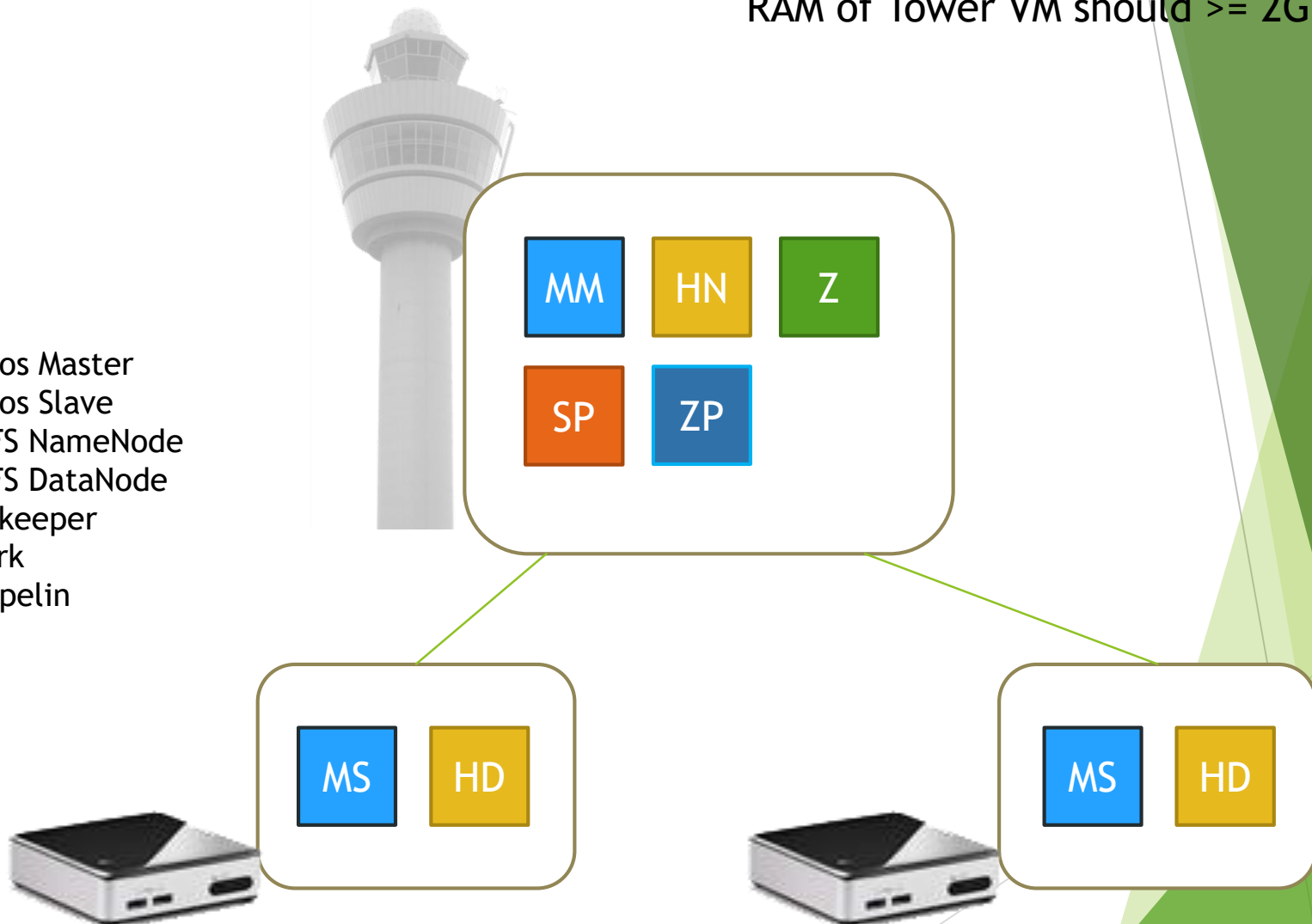
Support Spark



0. Cluster Overview

Prerequisite: Ubuntu 14.04 - 64bit.
(To install Java, see AppServer Lab.)
RAM of Tower VM should \geq 2GB.

- MM Mesos Master
- MS Mesos Slave
- HN HDFS NameNode
- HD HDFS DataNode
- Z Zookeeper
- SP Spark
- ZP Zeppelin



0. Preparation

- Install Java



Edit: Install JAVA JDK 8

```
$ sudo add-apt-repository ppa:webupd8team/java
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install oracle-java8-installer
```

If fail to install JAVA like

error: sudo add-apt-repository: command not found

Follow command

```
$ sudo apt-get install software-properties-common python-software-properties
```

```
$ sudo apt-get update
```

Do this for all tower and NUCs.

If you already installed Java 8 in previous labs, you can skip it.

0. Preparation

- Configure accounts

Do this for all tower and NUCs.

1. Set root password

- `sudo passwd`

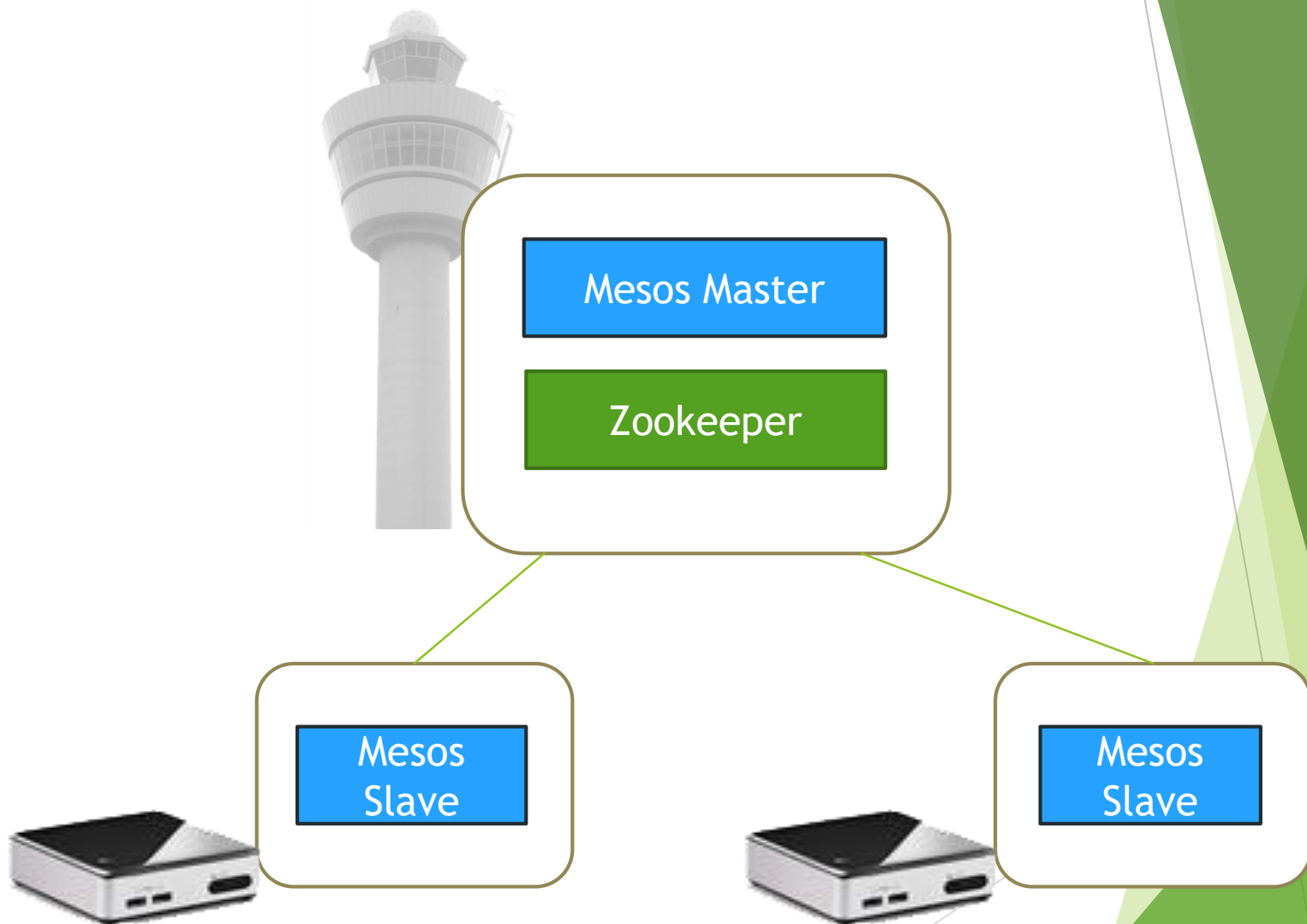
2. Create Hadoop account and login

- `sudo -s`
- `adduser hadoop`
- `adduser hadoop sudo`
- `su hadoop` `#login as 'hadoop'`



1. Apache Mesos

- Install



1. Apache Mesos

- Installation Procedure

1. Add Mesosphere repository
2. Install Mesos Master
3. Install Mesos Slave
4. Check on the Web UI

1. Apache Mesos

- Install: Add Mesosphere repository

Add the repository to Tower and NUCs.



```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv E56151BF
```

```
DISTRO=$(lsb_release -is | tr '[:upper:]' '[:lower:]')  
CODENAME=$(lsb_release -cs)
```

To check you correctly input, use echo command.

```
echo $DISTRO $CODENAME
```

```
ubuntu trusty
```

```
echo "deb http://repos.mesosphere.io/${DISTRO} ${CODENAME} main" | sudo  
tee /etc/apt/sources.list.d/mesosphere.list
```

```
sudo apt-get -y update
```


1. Apache Mesos

- Install: Mesos Master

```
sudo apt-get -y install mesos  
sudo reboot
```

```
sudo service mesos-slave stop  
echo manual | sudo tee /etc/init/mesos-slave.override  
echo <TOWER_IP_ADDR> | sudo tee /etc/mesos-master/ip  
echo <TOWER_IP_ADDR> | sudo tee /etc/mesos-master/hostname  
echo zk://<TOWER_IP_ADDR>:2181/mesos | sudo tee /etc/mesos/zk  
echo <YOUR_NAME> | sudo tee /etc/mesos-master/cluster  
sudo service zookeeper restart  
sudo service mesos-master restart
```

```
echo 1 | sudo tee /etc/zookeeper/conf/myid
```

YOUR_NAME: anything you want



1. Apache Mesos

- Install: Mesos Slave



```
sudo apt-get -y install mesos
sudo reboot
```

```
sudo service mesos-master stop
echo manual | sudo tee /etc/init/mesos-master.override
sudo service zookeeper stop
echo manual | sudo tee /etc/init/zookeeper.override
sudo apt-get -y remove --purge zookeeper
```

```
echo <NUC_IP_ADDR> | sudo tee /etc/mesos-slave/ip
```

```
echo <NUC_IP_ADDR> | sudo tee /etc/mesos-slave/hostname
```

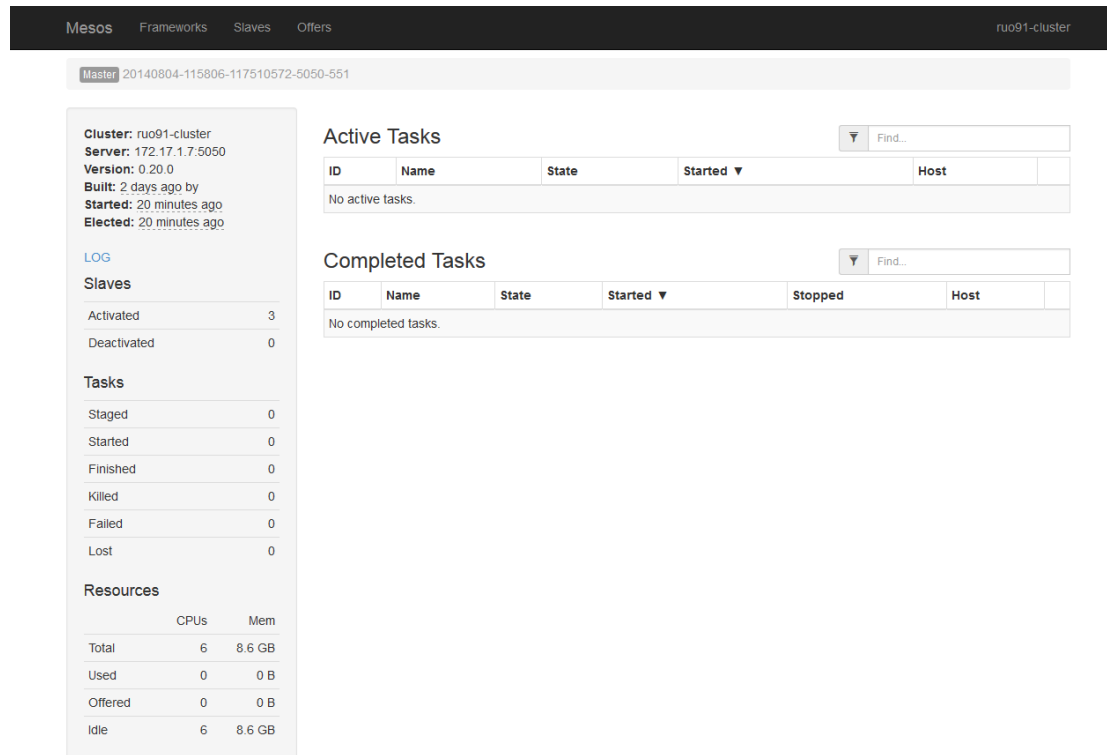
```
echo zk://<MASTER_IP_ADDR>:2181/mesos | sudo tee /etc/mesos/zk
sudo reboot
```

We installed Mesos Master on Tower, so
Tower IP address will be Master IP address.

1. Apache Mesos

- Check on the Web UI

In your web browser, go to
`http://<MASTER-IP-ADDR>:5050`



The screenshot displays the Apache Mesos Web UI for a cluster named 'ruo91-cluster'. The top navigation bar includes links for Mesos, Frameworks, Slaves, and Offers. The main content area is divided into several sections:

- Master:** 20140804-115806-117510572-5050-551
- Cluster Information:**
 - Cluster: ruo91-cluster
 - Server: 172.17.1.7:5050
 - Version: 0.20.0
 - Built: 2 days ago by
 - Started: 20 minutes ago
 - Elected: 20 minutes ago
- LOG**
- Slaves:**

Activated	Deactivated
3	0
- Tasks:**

Staged	Started	Finished	Killed	Failed	Lost
0	0	0	0	0	0
- Resources:**

	CPU	Mem
Total	6	8.6 GB
Used	0	0 B
Offered	0	0 B
Idle	6	8.6 GB
- Active Tasks:** A table with columns ID, Name, State, Started, and Host. It shows 'No active tasks.'
- Completed Tasks:** A table with columns ID, Name, State, Started, Stopped, and Host. It shows 'No completed tasks.'

Check the activated slaves and resources.



2. Apache Spark

- Install



```
wget http://mirror.apache-kr.org/spark/spark-1.6.1/spark-1.6.1-bin-hadoop2.6.tgz
tar xzf spark-1.6.1-bin-hadoop2.6.tgz
```

```
cd spark-1.6.1-bin-hadoop2.6/conf
cp spark-env.sh.template spark-env.sh
vi spark-env.sh
```

Edit: export MESOS_NATIVE_JAVA_LIBRARY=/usr/local/lib/libmesos.so
export MASTER=mesos://<MASTER_IP_ADDR>:5050

Test Spark: In Tower

```
cd ..
bin/pyspark

data = range(1, 10001)
distData = sc.parallelize(data)
distData.filter(lambda x: x < 10).collect()
```

Go to Mesos web UI and see Spark framework running.

3. Apache Zeppelin

- Install (on Mesos)



```
cd ..      # Go to home directory.  
wget http://mirror.apache-  
kr.org/incubator/zeppelin/0.5.6-incubating/zeppelin-  
0.5.6-incubating-bin-all.tgz
```

```
tar xzf zeppelin-0.5.6-incubating-bin-all.tgz
```

```
cd zeppelin-0.5.6-incubating-bin-all/conf  
cp zeppelin-env.sh.template zeppelin-env.sh  
vi zeppelin-env.sh
```

Edit: export MESOS_NATIVE_JAVA_LIBRARY=/usr/local/lib/libmesos.so
export MASTER=mesos://<MASTER_IP_ADDR>:5050
export SPARK_HOME=/home/<user>/spark-1.6.1-bin-hadoop2.6

```
cd ..  
bin/zeppelin-daemon.sh start
```

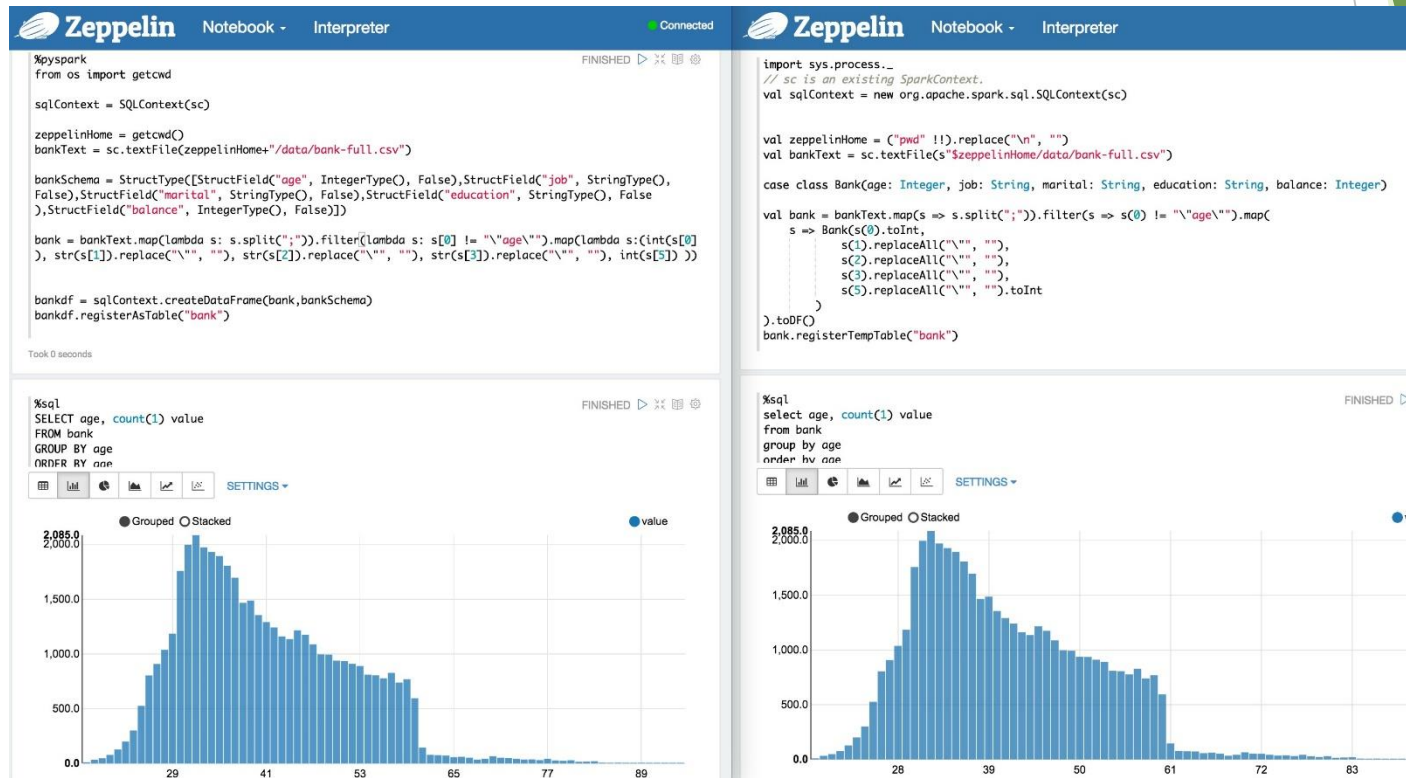
http://<Tower IP-ADDR>:8080

3. Apache Zeppelin

- Run Example



In Zeppelin tutorial, Press 'Run' button to test.



3. Apache Zeppelin

- Tip: Zeppelin Standalone mode

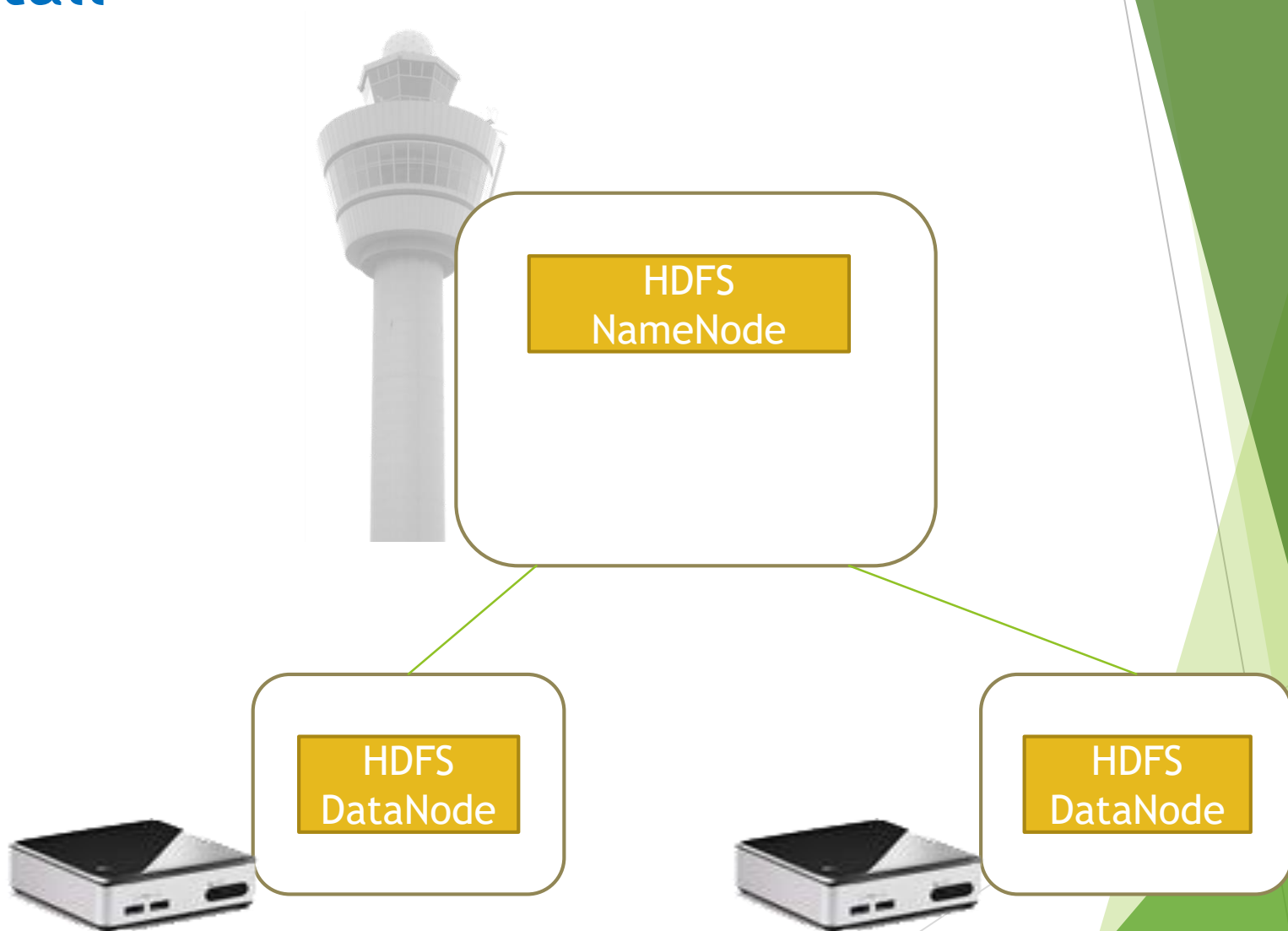
If you have trouble running Zeppelin on Mesos, you can run Zeppelin in standalone mode.

```
rm conf/zeppelin-env.sh  
bin/zeppelin-daemon.sh start  
#(or if daemon is already running, use 'restart' instead of 'start.')
```

<http://<IP-ADDR>:8080>

4. HDFS (Optional)

- Install



4. HDFS (Optional)

- Installation Procedure

1. Set hostnames
2. Configure accounts and SSH settings
3. Download and Unzip Hadoop
4. Configure HDFS
5. Start and test

4-1. HDFS

- Set Hostname

1. Registration host names

```
sudo vi /etc/hosts
```

Edit: Type IP addresses and hostnames of all nodes.

Ex)

192.168.0.1 tower

192.168.0.2 nuc1

192.168.0.3 nuc2

Do this for all tower and NUCs.



4-1. HDFS

- Configure accounts and SSH settings

3. Generate key (just press enter x 3) in tower and NUCs

- `ssh-keygen -t rsa`
- `cp /home/hadoop/.ssh/id_rsa.pub /home/hadoop/.ssh/authorized_keys`

4. Modify key permission

- `cd .ssh`
- `chmod 755 ~/.ssh`
- `chmod 644 authorized_keys`

5. Copy key from Tower to all NUCs

- `scp authorized_keys hadoop@<nuc_hostname>:.ssh/`

6. Try to login via SSH to check if you can login to NUC without password.



4-1. HDFS

- Download and Unzip Hadoop



1. Download and Unzip in Tower and NUCs.

- `cd ~`
- `wget http://mirror.apache-kr.org/hadoop/common/hadoop-2.7.2/hadoop-2.7.2.tar.gz`
- `tar -xvzf hadoop-2.7.2.tar.gz`
- `mv hadoop-2.7.2 hadoop`

4-1. HDFS

- Configuration



1. Go to the directory which contains configuration files.

- `cd hadoop/etc/hadoop`
- `hadoop-env.sh`, `core-site.xml`, `hdfs-site.xml`, `slaves`

2. `<Hadoop-env.sh>` file

Edit: `export JAVA_HOME=/usr/lib/jvm/java-8-oracle`

3. `<core-site.xml>` file

Edit: `<configuration>`
 `<property>`
 `<name>fs.defaultFS</name>`
 `<value>hdfs://hostname:9000/</value>`
 `</property>`
 `</configuration>`

This option must
be a hostname,
not IP address.

4-1. HDFS

- Configuration



4. <hdfs-site.xml> file

Edit: <configuration>
 <property>
 <name>dfs.replication</name>
 <value>3</value>
 </property>
 <property>
 <name>dfs.namenode.name.dir</name>
 <value>file:///home/hadoop/hadoop/namenode</value>
 </property>
 <property>
 <name>dfs.datanode.data.dir</name>
 <value>file:///home/hadoop/hadoop/datanode</value>
 </property>
</configuration>

5. <slaves> file: Add hostname or IP address of all NUCs, one per line.

Ex)
nuc1
nuc2

4-1. HDFS

- Configuration



6. In all Tower and NUCs, edit /etc/environment file.

```
vi /etc/environment
```

Add this line between the last word and ".

```
:/home/hadoop/hadoop/bin
```

Ex) `PATH="/usr/local/sbin:/usr/local/bin:
... :/sbin:/bin:/home/hadoop/hadoop/bin"`

7. Reboot.

4-1. HDFS

- Start and Test

1. Format NameNode.

```
bin/hdfs namenode -format
```

2. Start HDFS.

```
sbin/start-dfs.sh
```

3. Make a directory and upload a file to HDFS to check if it is working.

```
hadoop fs -mkdir /user  
hadoop fs -put <any_file> /user/  
hadoop fs -ls /user/
```

You can also see on the web:
http://<NameNode_IP>:50070



4-2. Apache Spark

- Configuration



In home directory

```
hadoop fs -put spark-1.6.1-bin-hadoop2.6.tgz /user/
```

```
cd spark-1.6.1-bin-hadoop2.6/conf
vi spark-env.sh
```

Edit: export MESOS_NATIVE_JAVA_LIBRARY=/usr/local/lib/libmesos.so
export MASTER=mesos://<MASTER_IP_ADDR>:5050
export SPARK_EXECUTOR_URI=hdfs://<TOWER_IP_ADDR>/user/spark-1.6.1-bin-hadoop2.6.tgz

Test Spark

```
cd ..
bin/pyspark

data = range(1, 10001)
distData = sc.parallelize(data)
distData.filter(lambda x: x < 10).collect()
```

Go to Mesos web UI and see Spark framework running.

4-3. Apache Zeppelin

- Configuration



```
cd zeppelin-0.5.6-incubating-bin-all/conf
cp zeppelin-env.sh.template zeppelin-env.sh
vi zeppelin-env.sh
```

Edit:

```
export MESOS_NATIVE_JAVA_LIBRARY=/usr/local/lib/libmesos.so
export MASTER=mesos://<MASTER_IP_ADDR>:5050
export SPARK_EXECUTOR_URI=hdfs://<HDFS_IP_ADDR>/user/spark-1.6.1-bin-hadoop2.6.tgz
```

```
cd ..
bin/zeppelin-daemon.sh start
```

<http://<Tower IP-ADDR>:8080>

Thank You for
Your Attention
Any Questions?

