과목명	빅데이터처리
담당교수	신현석 교수님
과제명	하둡 설치 보고서
학과	소프트웨어학과
학번	32151671
이름	박민혁
제출일자	2021-03-29

목차

I. 서론

1. 하둡이란

하둡을 배우기 전에, Big Data시대가 어떻게 됐는지에 대해서 알아야한다.
데이터가 폭발적으로 증가하였고, 데이터 유형이 구조적 데이터에서 비정형 즉 반구조적 데이터가 많아졌고, 데이터 특성이 다양해지고 복합적이며 실시간으로 변화한다. 그래서 이것들을 처리하기 위해 하둡이 필요하게 되었으며 하둡이 탄생하게 된 이유이다.

한명의 사용자가 하나의 DB 시스템에 요청을 하면 속도는 빠르다. 하지만 여러 명의 사용자가 요청을 하게 되면 부하가 걸릴 것이고 느려질 것이다. 그래서 사용자가 직접 DB에 요청을 하거나, 하둡을 이용해 요청을 하면 빨라질 것이다.

하둡이란 간단한 프로그래밍 모델을 사용하여 여러 대의 컴퓨터 클러스터에 대규모 데이터 셋을 분산 처리 할 수 있게 해주는 프레임 워크이다. 단일 서버에서 수천 대의 머신으로 확장 할 수 있도록 설계되었다. 일반적으로 하둡 파일시스템과 맵리듀스 프레임워크로 시작 되었으나, 여러 데이터 저장, 실행 엔진, 프로그래밍 및 데이터 처리 같은 하둡 생태계 전반을 포함하는 의미로 확장 발전 되었다.

하둡의 프로세싱 과정은 다음과 같다.
데이터 수집 -> 데이터 저장 -> 데이터 처리 -> 데이터 분석 -> 데이터 시각화이다. 하둡의 에코 시스템에 대해 설명하겠다.

1. 분산코디네이터

Zookeeper	분산환경에서 서버 간의 상호 조정이 필요한 다양한 서비스를 제공하는 시스템

2. 분산 리소스 관리

YARN	작업 스케줄링 및 클러스터 리소스 관리를 위한 프레임워크로 맵리듀스, 하이브, 임팔라, 스파크등 애플리케이션들은 얀에서 작업을 실행

3. 데이터 수집

Flume	비정형 데이터 수집
Sqoop	정형 데이터 수집

4. 데이터 저장

HBase	구글 빅데이터 기반으로 개발된 비관계형 데이터베이스이며 하둡 및 하둡 파일시스템 위에 빅테이블과 같은 기능을 제공
HDFS	애플리케이션 데이터에 대한 높은 처리량의 엑세스를 제공하는 분산 파일 시스템

Kafka	버퍼 기능이 있어 데이터를 저장

5. 데이터 처리

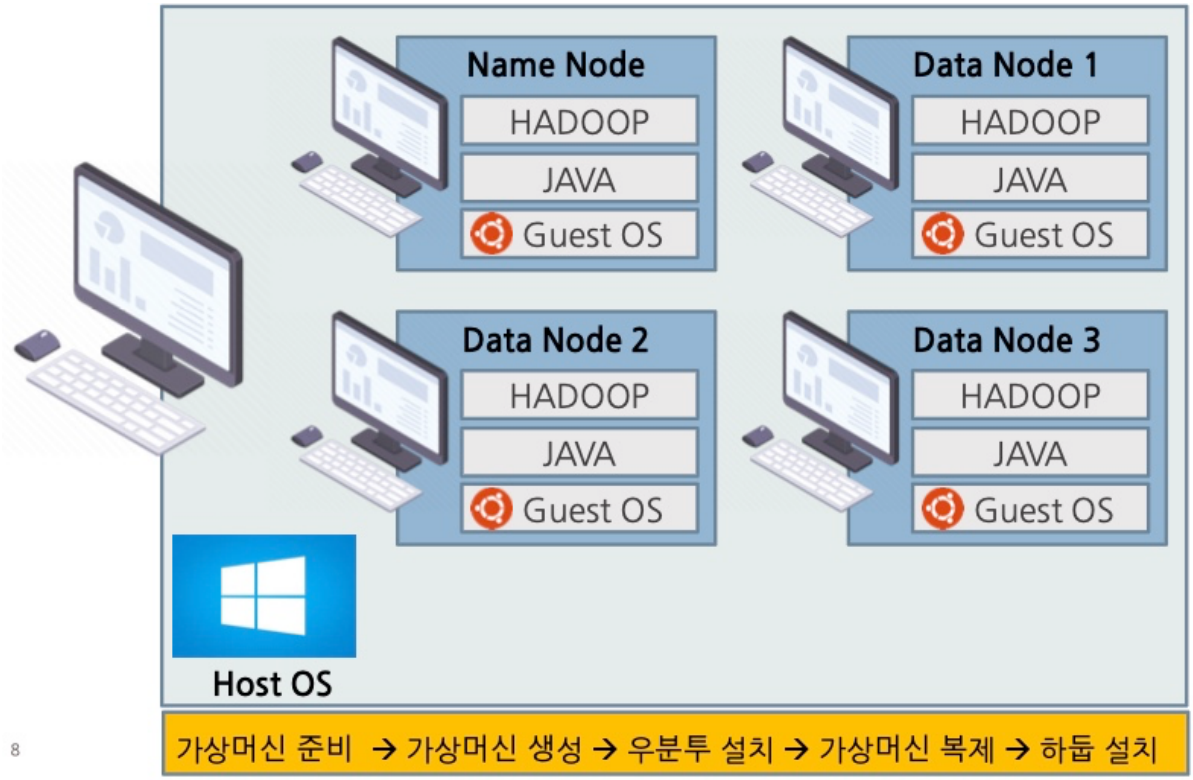
PIG	구조화, 비구조적 데이터 사용 가능
HIVE	구조화 된 데이터에서 사용 가능
Spark	메모리 기반 데이터 처리, 실시간으로 데이터를 처리(주식시장 그래프)
Mahout	분석 기계 학습에 필요한 알고리즘을 구축하기 위한 오픈소스 프레임워이며, 클러스터링, 추천시스템, 맵리듀스 등 머신러닝 알고리즘을 지원
STORM	실시간으로 데이터를 처리

6. 그 외

Apache Ambari	하둡을 모니터링 하여 상태를 확인
Oozie	하둡 관리

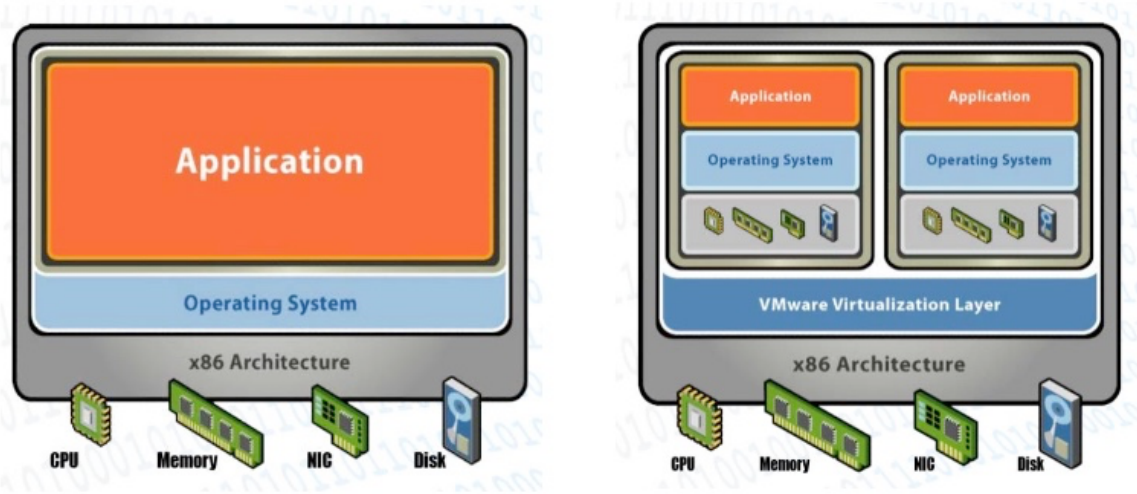
2. 가상 분산 모드란

Pseudo distributed mode, 즉 가상 분산 모드란 하나의 장비가 클러스터로 구성이 되어 있고, 모든 하둡(하둡 파일 시스템, 맵리듀스)과 관련 된 데몬이 이 장비에서만 실행 되는 환경을 말한다. 주로 맵리듀스 프로그램 개발 및 디버깅에 가장 많이 사용 되며 테스트 환경으로 적합한 모드이다.

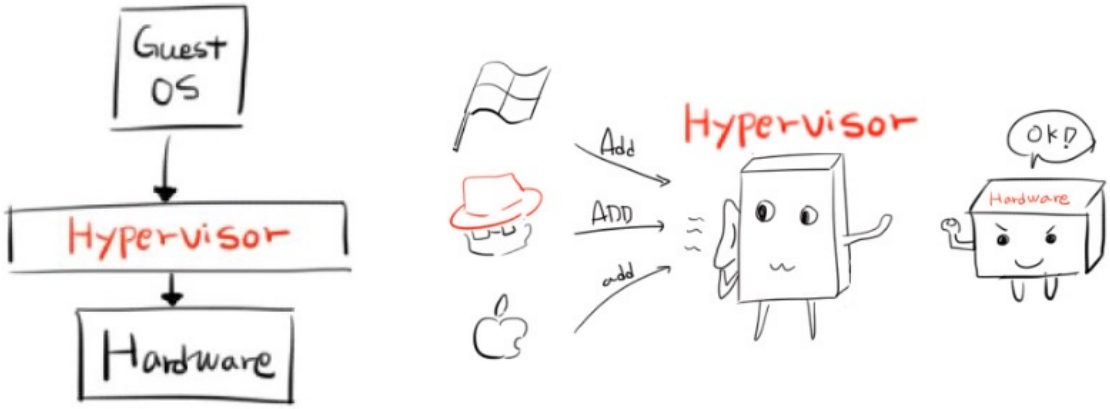


3. 가상 머신의 개념

가상 머신이란 물리적 컴퓨팅 환경을 소프트웨어로 구현한 것, 즉 컴퓨터를 에뮬레이션 하는 소프트웨어이다. 가상화는 물리적 서버에 하이퍼 바이저라고 하는 소프트웨어 계층을 추가한다.



즉 위에 사진으로 보면 PC안에 또다른 PC가 들어가 있는 형태이다.

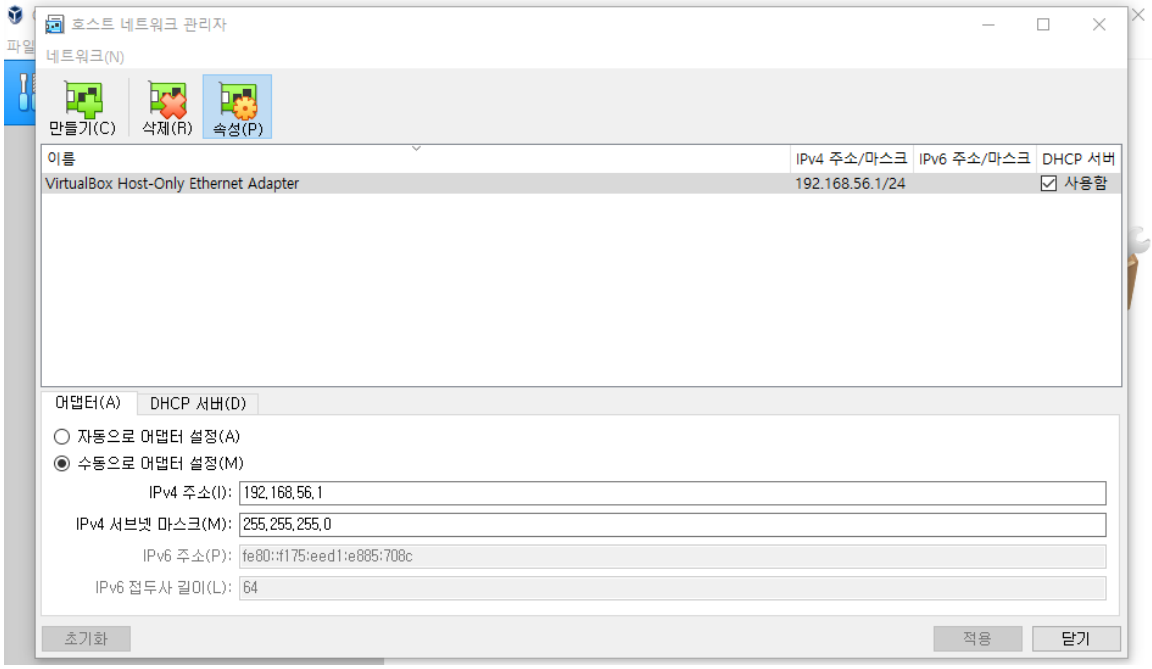


하이퍼 바이저의 역할은 Guest OS가 가상 머신이며 실제 컴퓨터 물리적 환경이랑 커뮤니케이션 하는 역할을 한다.

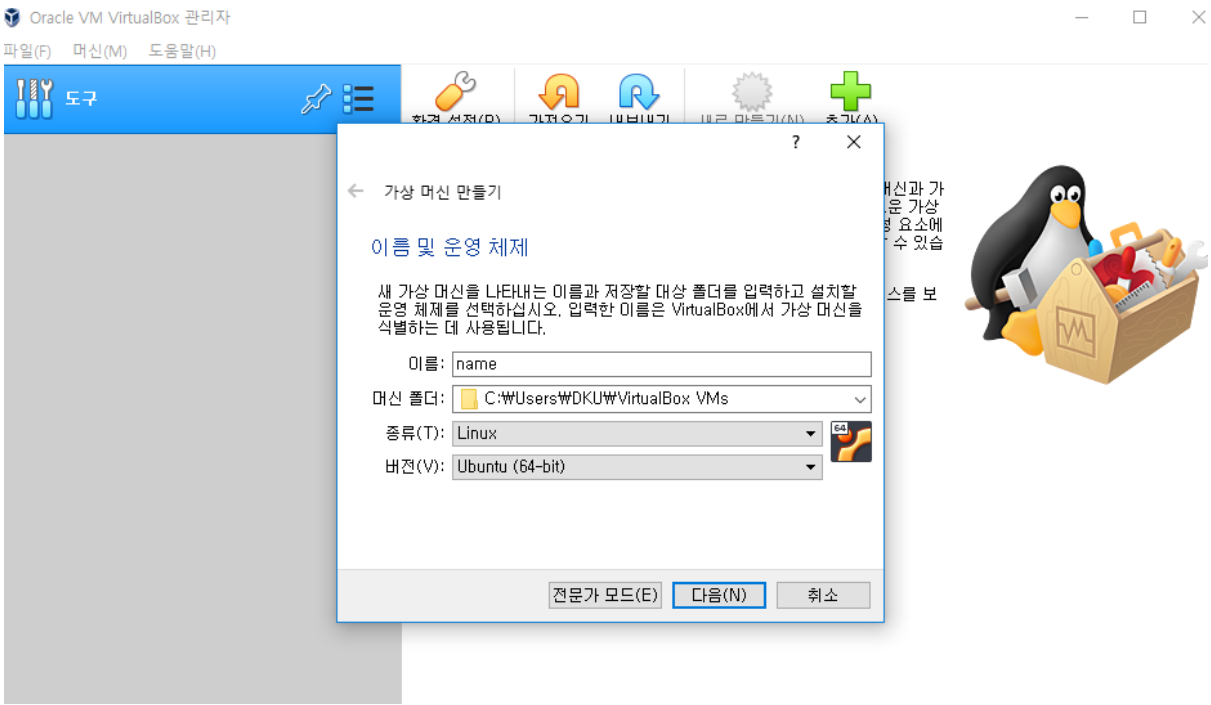
II. 본론

1. VirtualBox 설치

우선 VirtualBox.org에 접속한다. 접속하여 맞는 OS에 맞게 다운을 해준다. 다운을 해주면 고정 IP 설정을 한다.



위와 같이 고정 IP 설정을 마치면 name노드 설정을 한다.

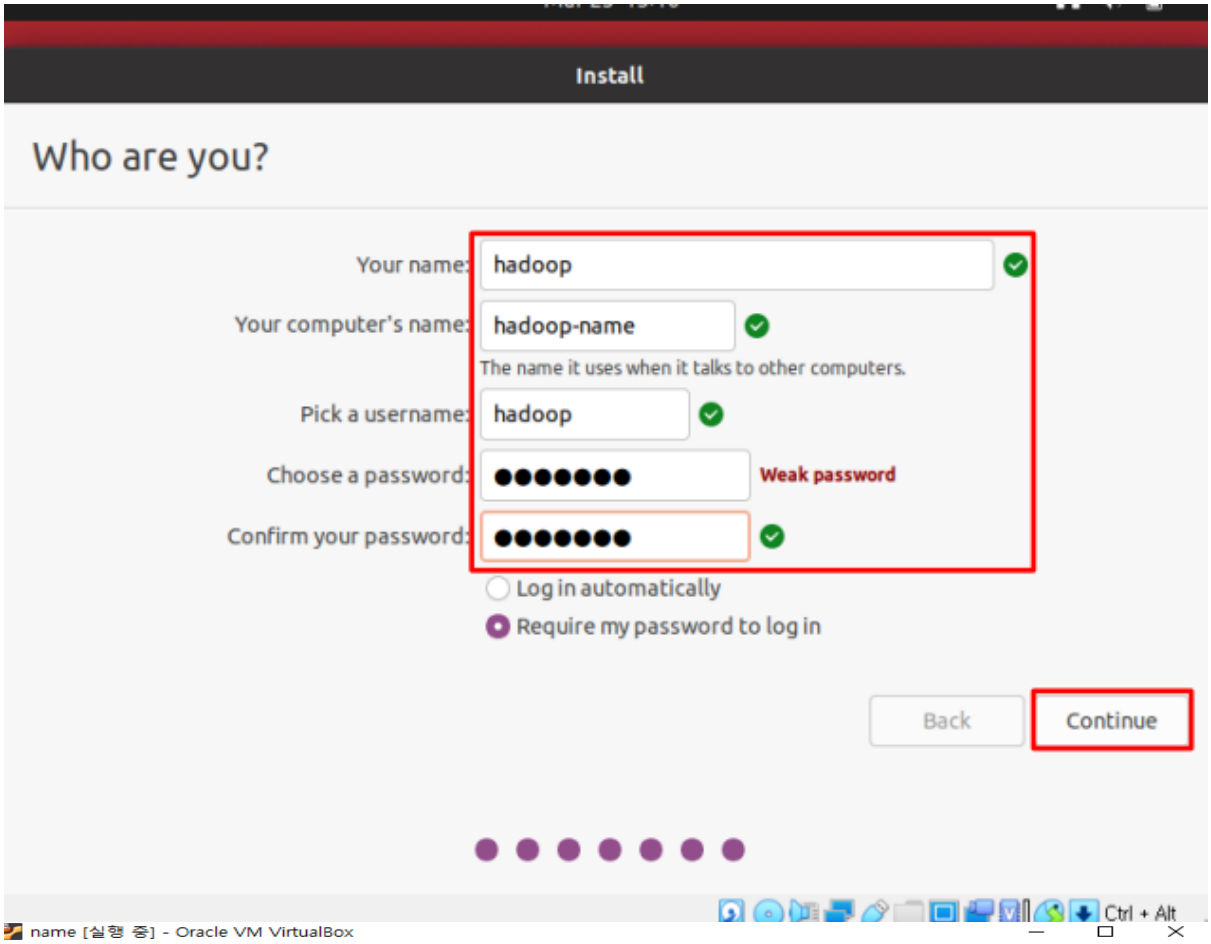


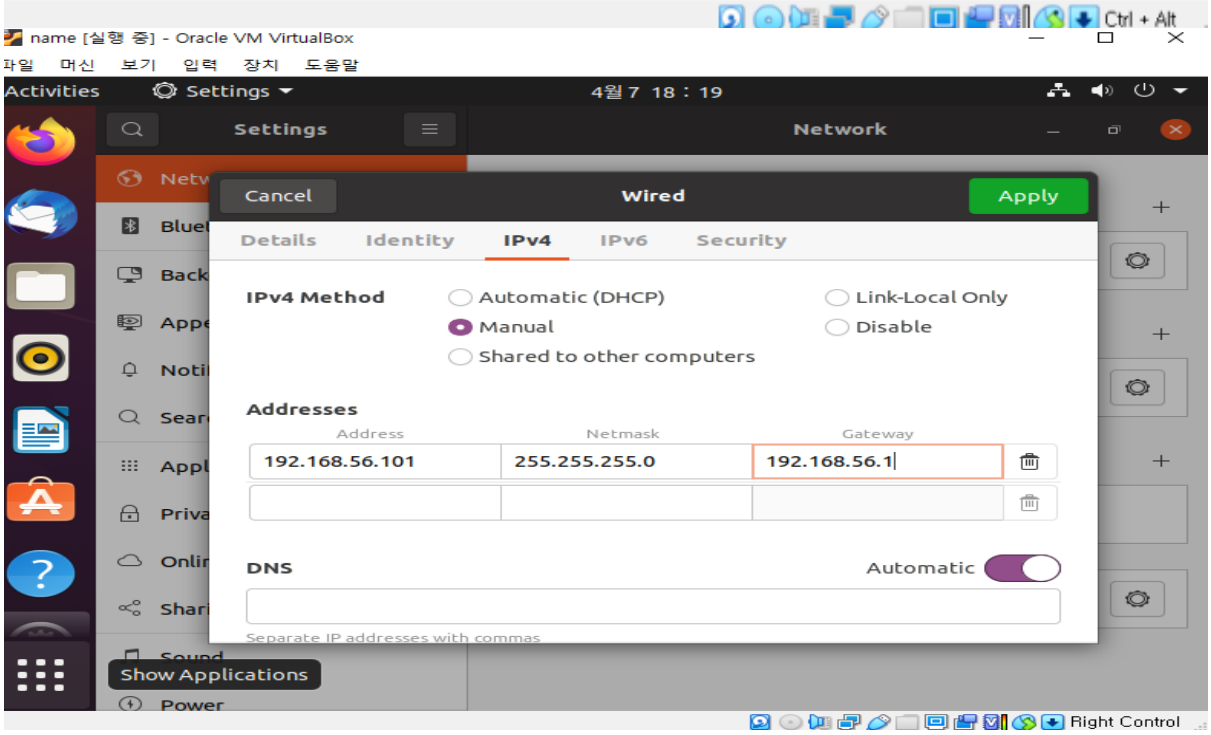
Name 노드 설정을 할 때 종류는 리눅스와 버전은 우분투 64비트 설정을 한다. 위와 같이 설정을 마치고 만든 다음에 네크워크 설정에서 어댑터 2를 들어가 고정IP 사용을 활성화 한다. 그리고 저장소에 들어가 다운 받았던 우분투 iso 파일을 삽입해 저장소를 만들어준다. Virtual box 설정은 이

것으로 끝낸다.

2. Ubuntu 설치 및 환경 구축

설정이 완료 된 Virtual box를 실행 시키고 다음과 같이 설정해준다.





설정을 마치고 가상 ip확인을 하고 ssh server를 설치한다.

```
hadoop@hadoop-name:~/Desktop$ sudo apt install ssh
Reading package lists... Done
```

설치가 끝난 ssh를 설정한다.

```
#Authentication:
LoginGraceTime 120
#PermitRootLogin without-password
PermitRootLogin yes
StrictModes yes
```

ssh 설정이 끝난후 java를 설치한다.

```
hadoop@hadoop-name:~/Desktop$ sudo service ssh restart
hadoop@hadoop-name:~/Desktop$ sudo apt-get install openjdk-8-jdk
```

Java 환경변수 설정을 해준다. (sudo vi /etc/profile)

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=$PATH:$JAVA_HOME/bin
export PATH
```

환경변수 설정후 java가 잘 깔렸는지 버전 확인을 해준다.

```
hadoop@hadoop-name:~/Desktop$ java -version
openjdk version "1.8.0_282"
OpenJDK Runtime Environment (build 1.8.0_282-8u282-b08-0ubuntu1~20.04-b08)
OpenJDK 64-Bit Server VM (build 25.282-b08, mixed mode)
hadoop@hadoop-name:~/Desktop$ ls
hadoop@hadoop-name:~/Desktop$ $JAVA_HOME/bin/javac -version
javac 1.8.0_282
hadoop@hadoop-name:~/Desktop$
```

Java 설정이 끝난후 host 파일을 수정하고 호스트 파일 수정을 한다.

```
192.168.56.101 master
192.168.56.102 slave1
192.168.56.103 slave2
192.168.56.104 slave3
```

Name 노드 설정이 끝난후 data1 ~ data3를 네임노드를 복제한다.

3. Hadoop 설치

복제를 마치면 모든 서버에 bigdata 디렉토리를 생성해준다. 디렉토리를 만들고 하둡을 다운받는다.

```
hadoop@hadoop-name:~/Desktop$ cd ~
hadoop@hadoop-name:~$ mkdir bigdata

hadoop@hadoop-name:~$
hadoop@hadoop-name:~$ cd bigdata
hadoop@hadoop-name:~/bigdata$ wget https://archive.apache.org/dist/hadoop/core/
hadoop-2.5.2/hadoop-2.5.2.tar.gz
```

하둡을 설치하고 설정하는 단계이다.

Hadoop-env.sh

```
23
24 # The java implementation to use.
25 export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
26
```

yarn-env.sh

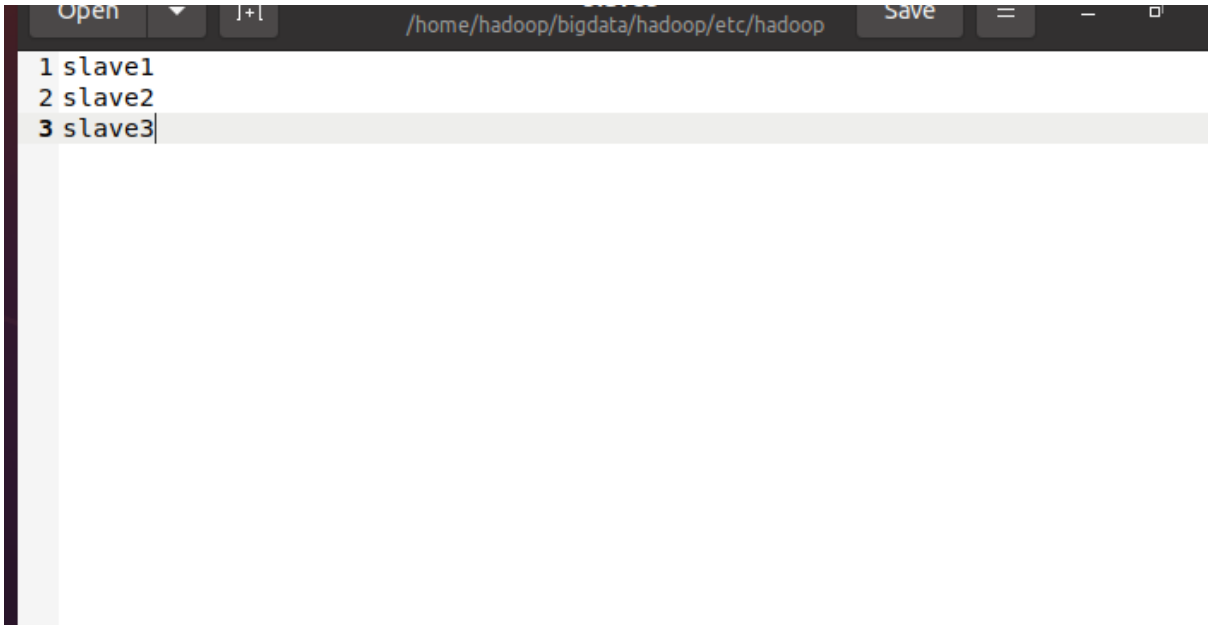
```
22 # some Java parameters
23 #export JAVA_HOME=/home/y/libexec/jdk1.6.0/
24 export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
25 if [ "$JAVA_HOME" != "" ]; then
26   #echo "run java in $JAVA_HOME"
27   JAVA_HOME=$JAVA_HOME
28 fi
29
```

Yarn-site.xml

```
<!-- Site specific YARN configuration properties -->
<property>
 <name>yarn.nodemanager.aux-services</name>
 <value>mapreduce_shuffle</value>
</property>
<property>
 <name>yarn.nodemanager.aux-
services.mapreduce.shuffle.class</name>
 <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>

</configuration>
```

Slaves



Core-site.xml

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://master:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/hadoop/bigdata/hadoop_tmp/tmp/</value>
</property>
</configuration>
```

Hdfs-site.xml

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://master:9000</value>
</property>
<property>
 <name>dfs.replication</name>
 <value>3</value>
</property>
<property>
 <name>dfs.namenode.name.dir</name>

<value>file:/home/hadoop/bigdata/hadoop_tmp/hdfs/namenode</value>
</property>
</configuration>
```

Mapred-site.xml

```
<configuration>
<property>
<name>mapreduce.job.tracker</name>
<value>master:5431</value>
</property>
<property>
<name>mapred.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

Ssh키 설정

```
hadoop@hadoop-name:~/bigdata/hadoop/etc/hadoop$ ssh-keygen -t rsa -f ~/.ssh/id_
rsa
Generating public/private rsa key pair.
Created directory '/home/hadoop/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/hadoop/.ssh/id_rsa
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:MZ0Dfs5xUxm4TuAclwdZ8bMukgjDaiGyvh/lcKyKWXU hadoop@hadoop-name
The key's randomart image is:
+---[RSA 3072]----+
|          .    .*=+ |
|         . oo.=.o. |
|          +o*+oo ..|
|       .   *o+o.  o|
| . ...E+S oo     . |
| .o...o o . o .    |
|.o+. o    . o . .  |
|==. o         . .  |
|*+o.              |
+----[SHA256]-----+
```

```
hadoop@hadoop-name:~/bigdata/hadoop/etc/hadoop$ cat ~/.ssh/id_rsa.pub >> ~/.ssh
/authorized_keys
```

공개키 배포 나머지 slave2, slave3도 동일하게 배포

```
hadoop@hadoop-name:~/bigdata/hadoop/etc/hadoop$ ssh-copy-id -i $HOME/.ssh/id_rs
a.pub hadoop@slave1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hadoop/.ss
h/id_rsa.pub"
The authenticity of host 'slave1 (192.168.56.102)' can't be established.
ECDSA key fingerprint is SHA256:kHLjQV7CuPVT++3NUEbBbRikeEbQUh6ZSSSTE/SHSm0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
 out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are promp
ted now it is to install the new keys
hadoop@slave1's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'hadoop@slave1'"
and check to make sure that only the key(s) you wanted were added.

hadoop@hadoop-name:~/bigdata/hadoop/etc/hadoop$
```

Hadoop 디렉토리를 각 노드에 배포 나머지 slave2, slave3도 동일하게 배포

```
hadoop@hadoop-name:~/bigdata/hadoop/etc/hadoop$ sudo rsync -avxP /home/hadoop/b
igdata/hadoop/ hadoop@slave1:/home/hadoop/bigdata/hadoop
```

환경변수 등록(name, slave1, slave2, slave3 동일하게 등록)

```
119 export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
120 export HADOOP_HOME=/home/hadoop/bigdata/hadoop
121 export HADOOP_MAPRED_HOME=$HADOOP_HOME
122 export HADOOP_COMMON_HOME=$HADOOP_HOME
123 export HADOOP_HDFS_HOME=$HADOOP_HOME
124 export YARN_HOME=$HADOOP_HOME
125 export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
126 export YARN_CONF_DIR=$HADOOP_HOME/etc/hadoop
127 export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
128 export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
129 export PATH=$PATH:$HADOOP_HOME/bin
130 export PATH=$PATH:$HADOOP_HOME/sbin
131 export PATH=$PATH:$JAVA_HOME/bin
132 export PATH
133
```

데몬실행

```
hadoop@hadoop-name:~/Desktop$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
21/04/07 21:54:17 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Starting namenodes on [master]
The authenticity of host 'master (192.168.56.101)' can't be established.
ECDSA key fingerprint is SHA256:kHLjQV7CuPVT++3NUEbBbRikeEbQUh6ZSSSTE/SHSm0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
master: Warning: Permanently added 'master,192.168.56.101' (ECDSA) to the list
of known hosts.
master: starting namenode, logging to /home/hadoop/bigdata/hadoop/logs/hadoop-h
adoop-namenode-hadoop-name.out
slave3: starting datanode, logging to /home/hadoop/bigdata/hadoop/logs/hadoop-h
adoop-datanode-hadoop-data3.out
slave2: ssh: connect to host slave2 port 22: No route to host
slave1: ssh: connect to host slave1 port 22: No route to host
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 SHA256:kHLjQV7CuPVT++3NUEbBbRikeEbQUh6ZSSSTE/SHSm0.
```

실행확인

```
hadoop@hadoop-name:~/Desktop$ jps
2481 SecondaryNameNode
2595 ResourceManager
2279 NameNode
2844 Jps
```

Hadoop 종료

```
hadoop@hadoop-name:~/Desktop$ stop-all.sh
This script is Deprecated. Instead use stop-dfs.sh and stop-yarn.sh
21/04/07 21:55:38 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
Stopping namenodes on [master]
master: stopping namenode
slave3: stopping datanode
slave1: ssh: connect to host slave1 port 22: No route to host
slave2: ssh: connect to host slave2 port 22: No route to host
Stopping secondary namenodes [0.0.0.0]
0.0.0.0: stopping secondarynamenode
21/04/07 21:56:00 WARN util.NativeCodeLoader: Unable to load native-hadoop libr
ary for your platform... using builtin-java classes where applicable
stopping yarn daemons
stopping resourcemanager
slave3: no nodemanager to stop
slave2: ssh: connect to host slave2 port 22: No route to host
slave1: ssh: connect to host slave1 port 22: No route to host
no proxyserver to stop
hadoop@hadoop-name:~/Desktop$
```

III. 결론

1. 설치 시 어려웠던 점

설치 시 어려웠던 점은 많다. 우선 쓰는 노트북이 m1칩 기반 맥북이다. Virtual box에서 m1칩은 아직 지원하지 않았고, 지원하지 않는 프로그램을 깔려다 보니 오류가 났고 오류 수정을 위해 시간을 많이 투자했다. 하지만 찾아 본 결과 지원하지 않는 프로그램이었고, 지원하지 않다 보니 다른 가상 머신을 찾아야 했다. 그러다 찾은 것이 vmware 가상 머신인데, 이것은 유료이다. 무료버전으로 나와 있는 것이 있는데 그것은 인텔기반 맥북만 지원하는 것이었다. 그래서 어쩔 수 없이 학교 강의실에서 실습을 진행했다. 실습을 진행할 때 어려웠던 점은 우선 컴퓨터 메모리가 작아 data 노드를 전부다 키지 못하고 하나씩 밖에 키지 못해 설정할 때 하나를 설정하고 하나를 다시 켜서 설정하고 이런 식으로 하다 보니 시간이 더 오래 걸렸다. 그런 부분을 제외하고는 설치하는데 순차적으로 진행하면 어려운 점은 없었던 것 같다.

2. 이해하기 어려웠던 부분

이해하기 어려웠던 부분은 딱히 없고, 수정한 파일들이 무슨 역할을 하는지 자세한 설명이 ppt에 나와있었으면 좋았는데 강의에서 설명해주셔서 딱히 상관은 없었다.

3. 문제 해결

문제 해결 이라고는 민망하지만 어찌 됐든 노트북에 깔리지 않아 집에 있는 데스크탑을 연결해서 시도했었다. 데스크탑이 오래 된 것이라 조금 느렸지만 윈도우 기반이었고 설치를 했다. 하지만 문제점이 컴퓨터가 가상화 지원 설정이 안 돼있었고 인터넷에 검색해보니 바이오스에 들어가 가상화 설정을 바꾸어 주라는 해답이 있었다. 그래서 컴퓨터 부팅을 하고 바이오스에 들어가 설정을 바꾸어 주었는데도 계속 오류가 떴다. 그래서 학교 가는 김에 학교에서 실습을 진행했다. 아 문제가 해결이 안 된 부분이 있는데 data 노드를 세 개 다 키지 못하는 점이다. 메모리를 적게 사용해도 4개를 한번에 켜놓으면 계속 컴퓨터가 팅기는 현상이 발생했다. 메모리를 또 너무 적게 만들면 가상 머신 자체가 너무 느려서 아무것도 할 수 없었다. 이 부분은 좀 더 생각해야겠다.