

Big Data Architecture Building

DONG HO SEO | MIN JAE JIN | KEON HOON LEE | HYUN KYUNG KIM

2018. 11. 23 (Fri)

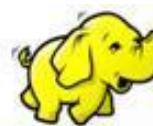
Contents

- 01**  Architecture
- 02**  Requirements
- 03**  Service Usage Process
- 04**  Demo
- 05**  Errors & Corrections
- 06**  Appendix

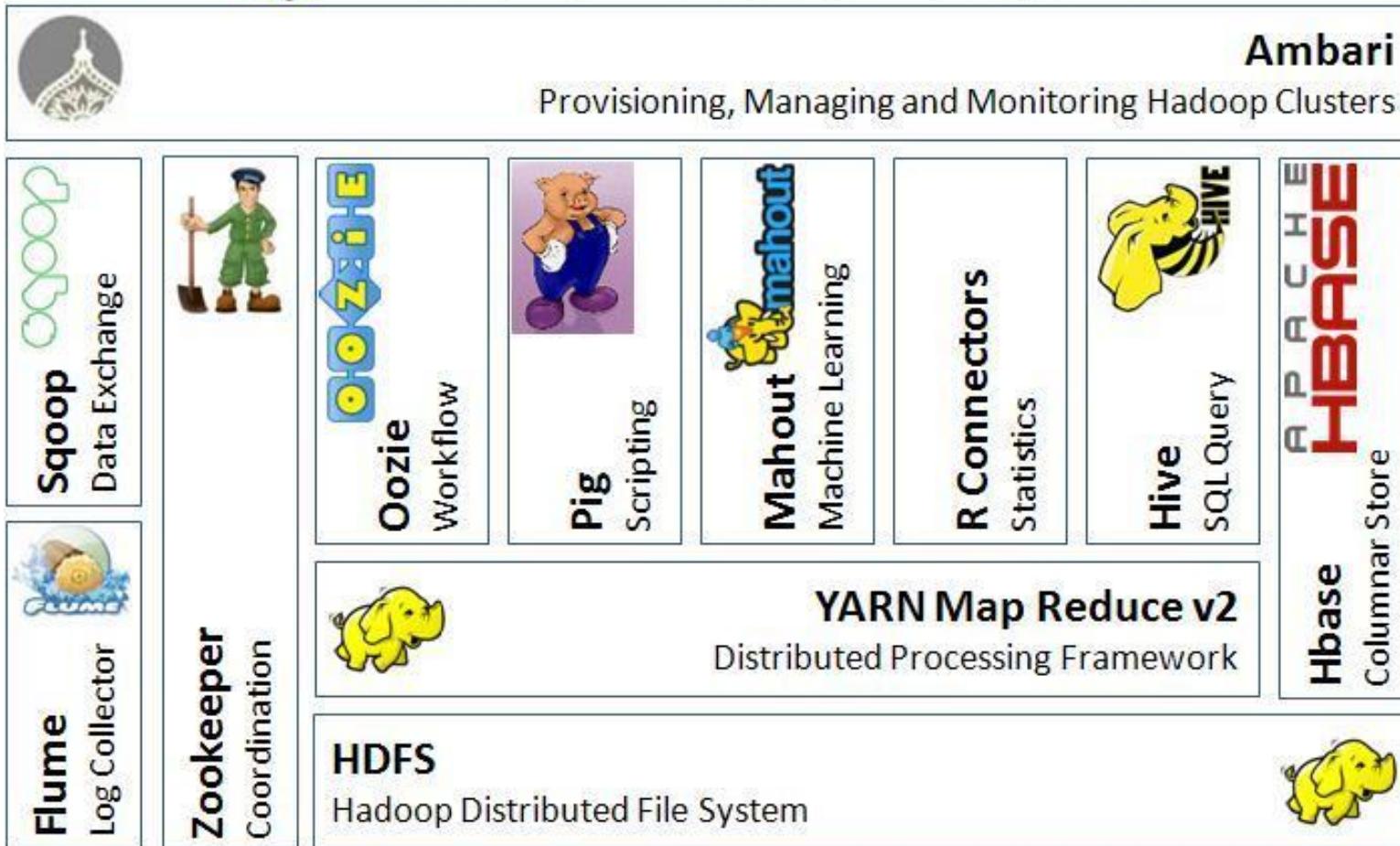
01 | Architecture

BY DELIGHT

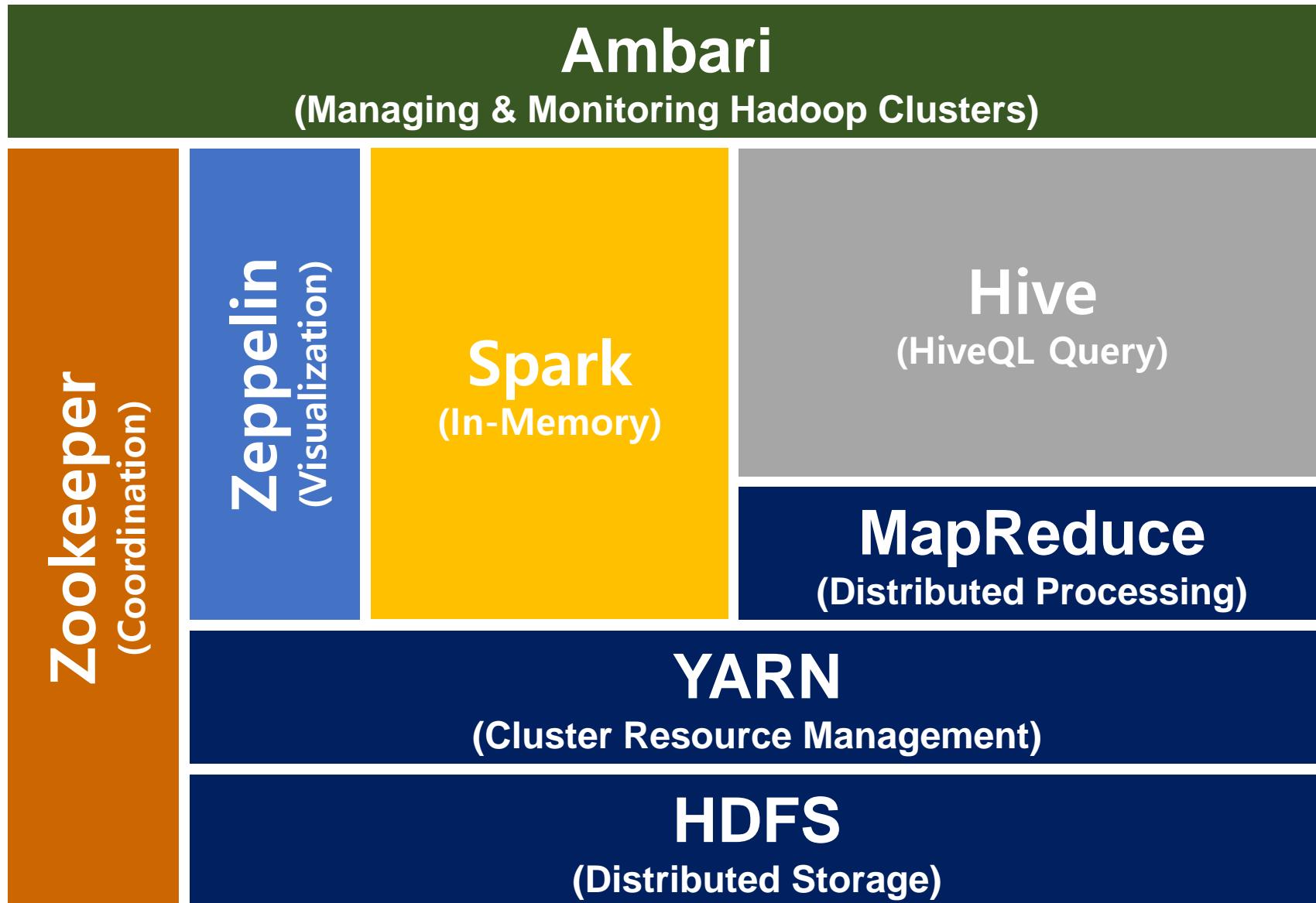
1.1 General Architecture



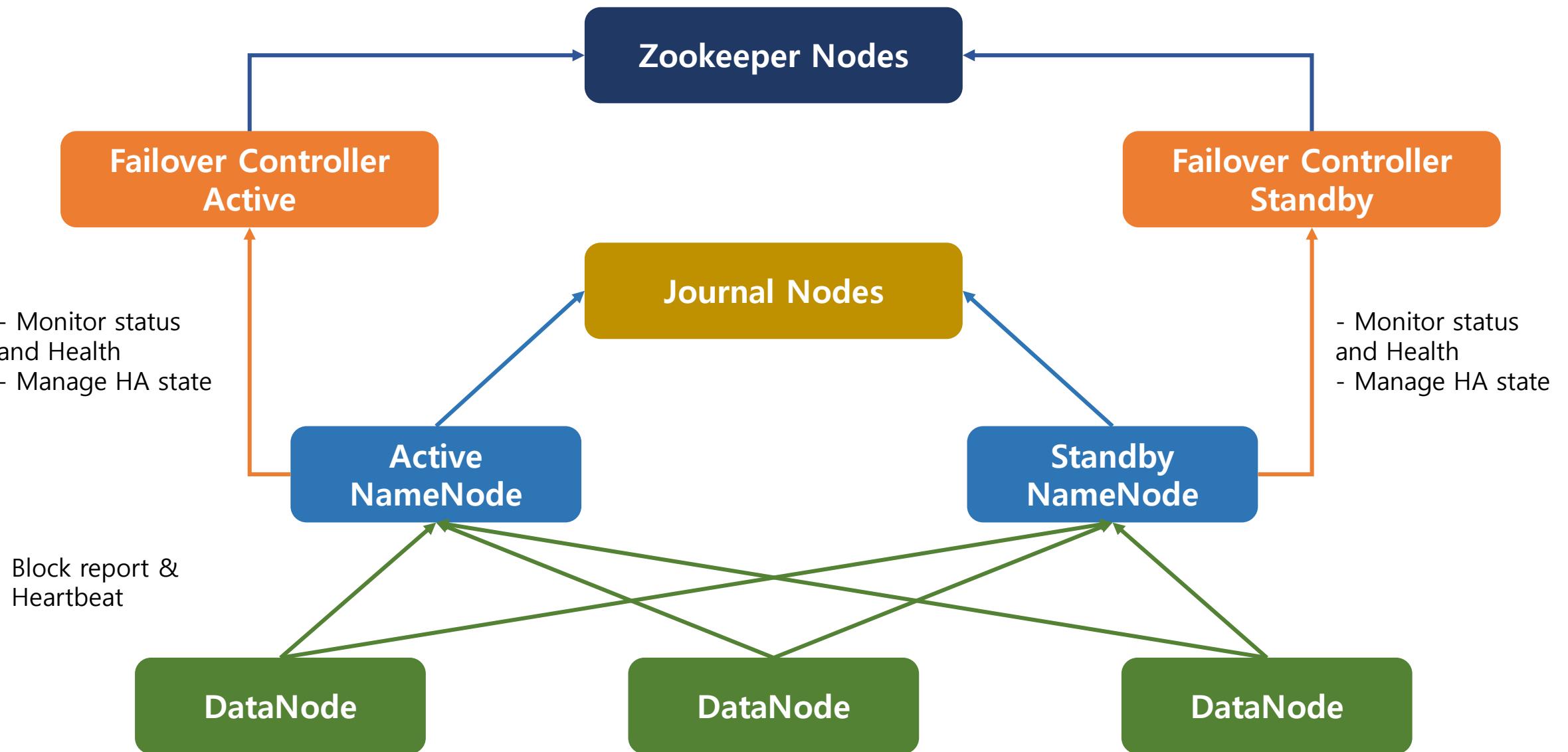
Apache Hadoop Ecosystem



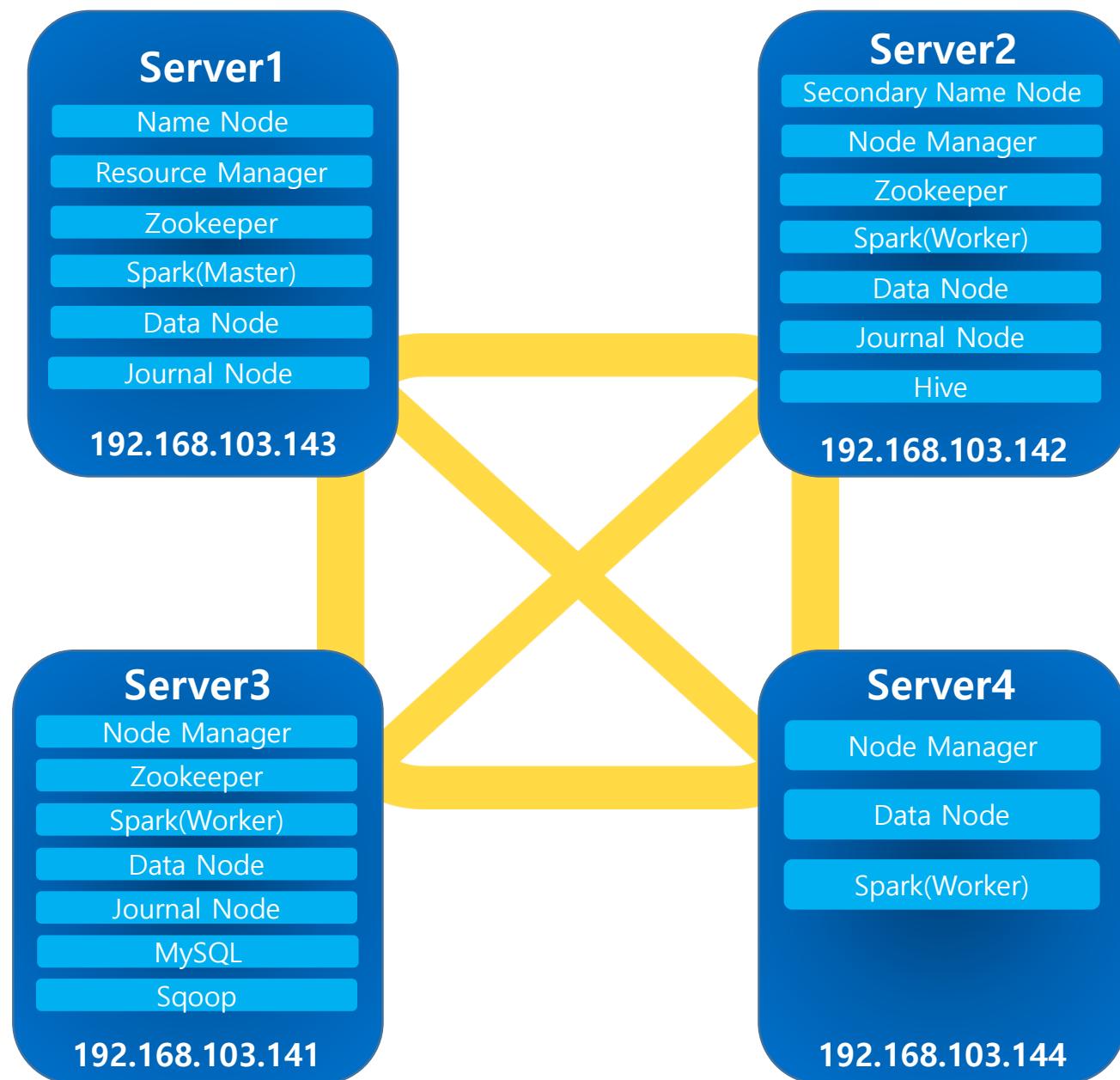
1.2 Our Architecture



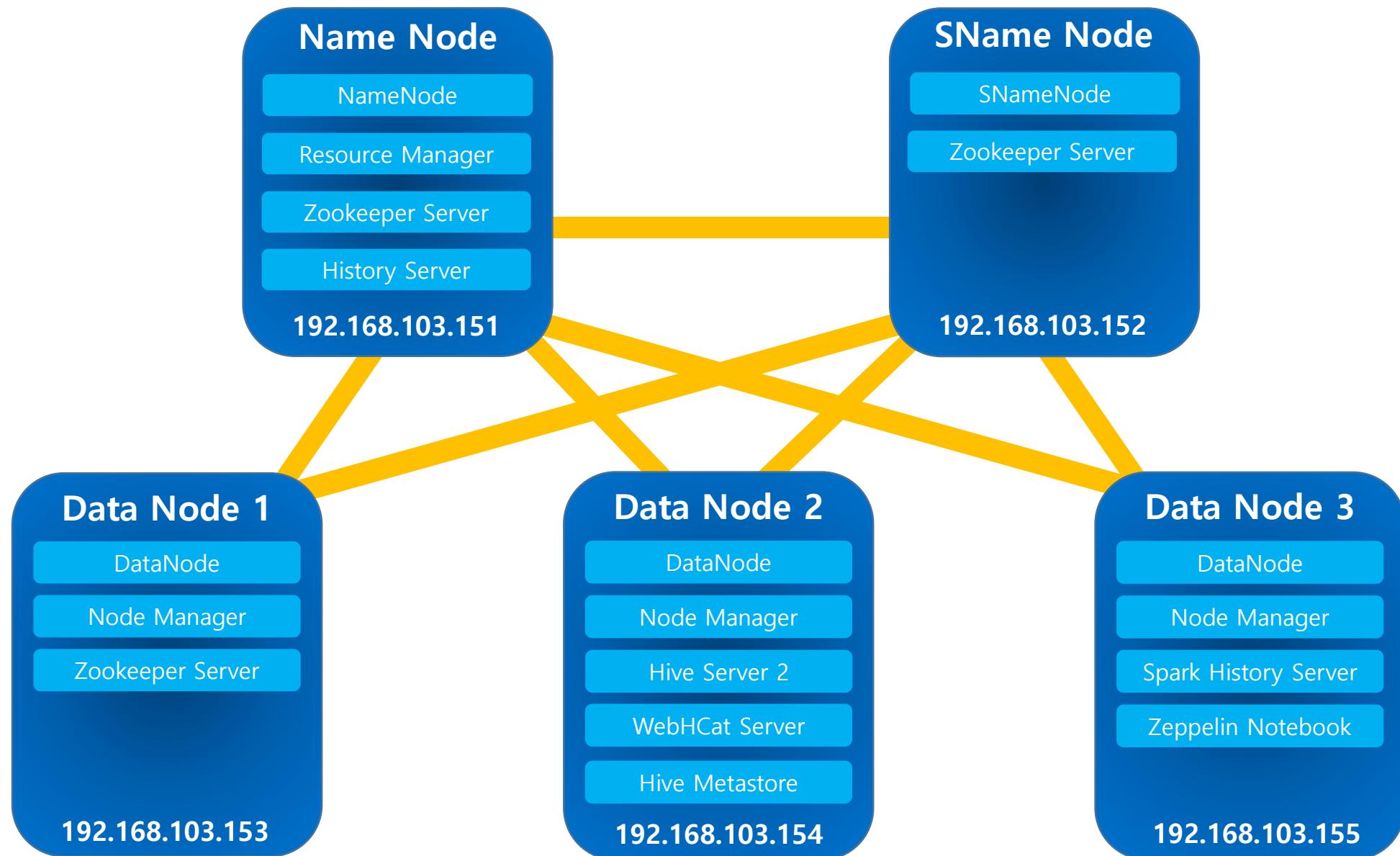
1.3 System Diagram (HDFS High Availability)



1.3.1 System Diagram (HDFS High Availability)



1.4 System Diagram (Ambari)



02 | Requirements

BY DELIGHT

2.1 Test Requirements

3개 이상의 DataNode로 구성된 Cluster를 구성하는
HDFS를 만들 것

- YARN, MR, Zookeeper가 service로 존재
- Data Block Size는 32 MB
- Block Replication은 2
- Namenode와 hive service는 각각 다른 host에서 구동

각 host는 다음 사양 만족

- Host의 Disk Size는 48 ~ 64 GB로 설정
- Host의 CPU Core는 2 ~ 4개로 설정
- Host의 Ram Size는 4.8 GB 이상으로 설정
- HDP 설치 시 SSH private key 입력을 통한 인증 등록



Hive, Spark & Zeppelin notebook 서비스 제공

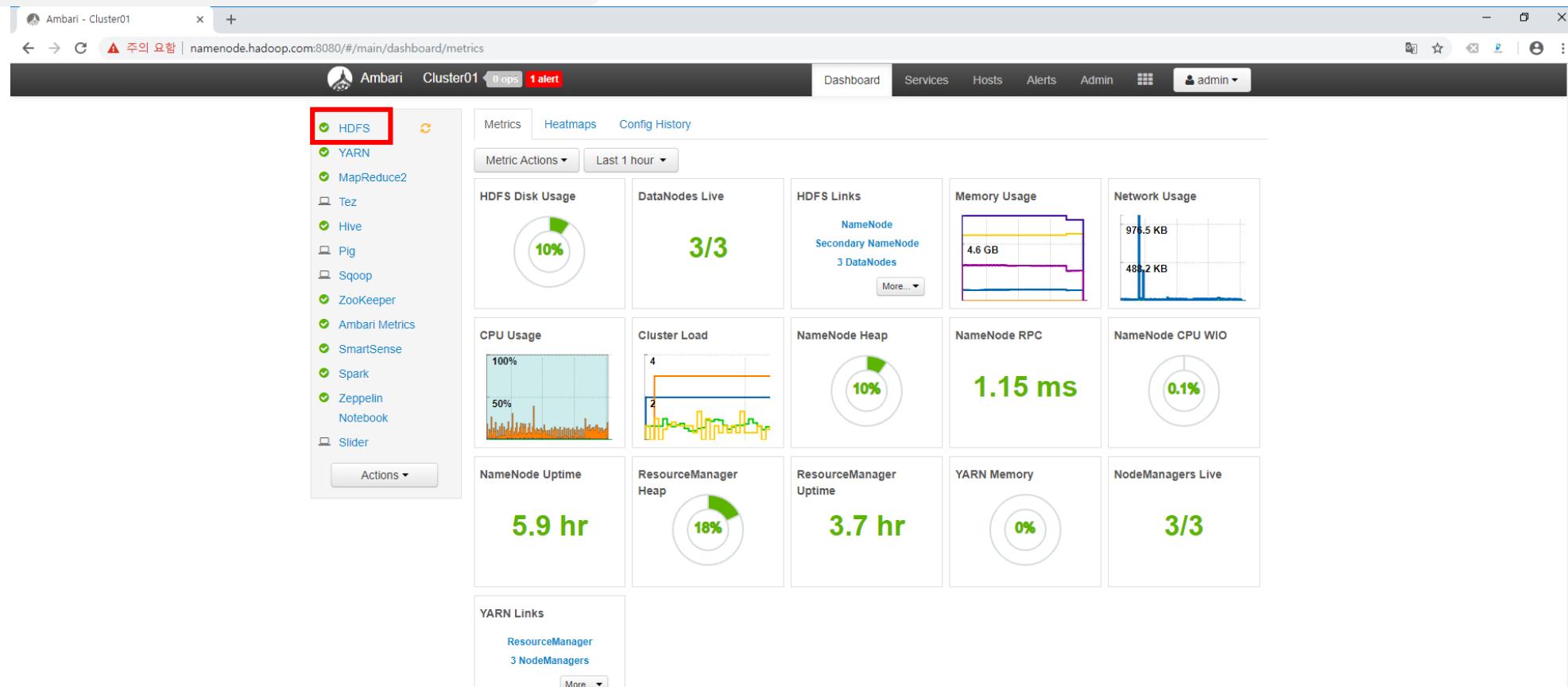
- Hive metastore는 hive server와 동일한 host에 존재
- Hive JDBC 제공
- Data가 존재하는 table 3개 이상 생성

가상 브릿지 네트워크 통신 구성

- Root는 패스워드가 없이 접근 가능
- 클러스터는 외부에서 접근 가능하도록 가상 브릿지 네트워크를 이용한 통신
- 방화벽 서비스는 disable
- 서비스 연결시 host명 return 문제를 해결

2.2 Requirements in Detail (Ambari)

Data block size 설정 확인



2.2 Requirements in Detail (Ambari)

Data block size 설정 확인

The screenshot shows the Ambari interface for Cluster01. The top navigation bar indicates 0 ops and 1 alert. The left sidebar lists various services: HDFS (selected), YARN, MapReduce2, Tez, Hive, Pig, Swoop, ZooKeeper, Ambari Metrics, SmartSense, Spark, Zeppelin Notebook, and Slider. The main content area has tabs for Summary, Heatmaps, and Configs, with Configs highlighted and a red box drawn around it. A yellow banner at the top states "Restart Required: 1 Component on 1 Host" with a "Restart" button. The Summary section provides details about NameNodes, DataNodes, JournalNodes, NFSGateways, and various metrics like disk usage and upgrade status. The Metrics section displays real-time charts for NameNode GC count, GC time, Connection Load, Heap memory, Host Load, RPC latency, Failed disk volumes, Corrupted blocks, Under-replicated blocks, and HDFS space utilization.

Summary

- NameNode: Started, No alerts
- SNameNode: Started, No alerts
- DataNodes: 3/3 Started
- DataNodes Status: 3 live / 0 dead / 0 decommissioning
- JournalNodes: 0/0 JournalNodes Live
- NFSGateways: 0/0 Started
- NameNode Uptime: 5.93 hours
- NameNode Heap: 49.2 MB / 1011.3 MB (4.9% used)
- Disk Usage (DFS Used): 3.2 GB / 93.6 GB (3.44%)
- Disk Usage (Non DFS Used): 6.5 GB / 93.6 GB (6.95%)

Metrics

- NameNode GC count: 1 ms
- NameNode GC time: 1 ms
- NN Connection Load: 10 ms
- NameNode Heap: 1000 MB
- NameNode Host Load: 100 %
- NameNode RPC: 10 ms
- Failed disk volumes: 0
- Blocks With Corrupted Replicas: 2
- Under Replicated Blocks: 0
- HDFS Space Utilization: 4%

2.2 Requirements in Detail (Ambari)

Data block size 설정 확인

The screenshot shows the Ambari web interface for a cluster named 'Cluster01'. The left sidebar lists various services: HDFS, YARN, MapReduce2, Tez, Hive, Pig, Sqoop, ZooKeeper, Ambari Metrics, SmartSense, Spark, and Zeppelin. The 'HDFS' service is selected. The main content area displays the 'Configs' tab for the HDFS service. A prominent message at the top states 'Restart Required: 1 Component on 1 Host' with a 'Restart' button. Below this, the 'Manage Config Groups' section shows a dropdown menu set to 'block', which is highlighted with a red box. A list of configuration versions is shown, with version V8 selected. At the bottom, there are 'Discard' and 'Save' buttons. Navigation tabs for 'Settings' and 'Advanced' are also visible.

2.2 Requirements in Detail (Ambari)

Data block size 설정 확인

namenode.hadoop.com:8080/#/main/services/HDFS/configs

Ambari Cluster01 0 ops 1 alert

Dashboard Services Hosts Alerts Admin admin

HDFS YARN MapReduce2 Tez Hive Pig Sqoop ZooKeeper Ambari Metrics SmartSense Spark Zeppelin

Summary Heatmaps Configs Quick Links

Restart Required: 1 Component on 1 Host

Group Default (5) Manage Config Groups

block

V8 admin 4 days ago HDP-2.6 ✓ V7 admin 4 days ago HDP-2.6 V6 admin 5 days ago HDP-2.6 V5 admin 5 days ago HDP-2.6 V4 admin 5 days ago HDP-2.6 V3 admin 5 days ago HDP-2.6

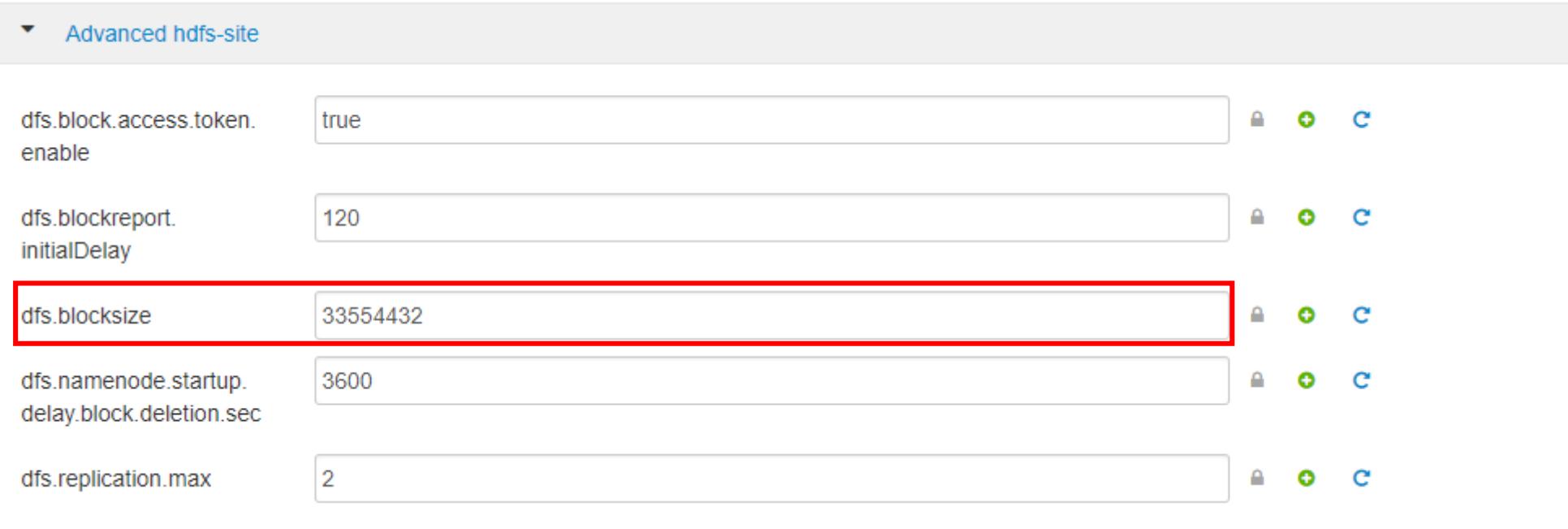
V8 ✓ admin authored on Fri, Nov 16, 2018 19:29

Discard Save

Settings Advanced

2.2 Requirements in Detail (Ambari)

Data block size 설정 확인



The screenshot shows the Ambari configuration interface for the 'Advanced hdfs-site' section. It displays several configuration parameters with their current values:

- dfs.block.access.token.enable: true
- dfs.blockreport.initialDelay: 120
- dfs.blocksize: 33554432 (This field is highlighted with a red border.)
- dfs.namenode.startup.delay.block.deletion.sec: 3600
- dfs.replication.max: 2

Each parameter has a lock icon, a plus sign icon, and a circular arrow icon to its right.

2.2 Requirements in Detail (Ambari)

Block Replication 설정 확인

The screenshot shows the Ambari interface for managing HDFS configurations. The left sidebar lists various services: HDFS, YARN, MapReduce2, Tez, Hive, Pig, Sqoop, ZooKeeper, Ambari Metrics, SmartSense, Spark, and Zeppelin. The 'HDFS' service is selected. The top navigation bar shows the cluster name 'Cluster01' with 0 ops and 1 alert.

The main content area displays the 'Configs' tab for HDFS. A yellow banner at the top indicates 'Restart Required: 1 Component on 1 Host'. Below this, a section titled 'Manage Config Groups' shows a dropdown menu set to 'block'. A list of configuration versions is shown, with version V8 highlighted. The status bar at the bottom shows 'admin authored on Fri, Nov 16, 2018 19:29'.

| Version | Author | Authored On | Host |
|---------|--------|-------------|---------|
| V8 | admin | 4 days ago | HDP-2.6 |
| V7 | admin | 4 days ago | HDP-2.6 |
| V6 | admin | 5 days ago | HDP-2.6 |
| V5 | admin | 5 days ago | HDP-2.6 |
| V4 | admin | 5 days ago | HDP-2.6 |
| V3 | admin | 5 days ago | HDP-2.6 |

At the bottom, there are 'Discard' and 'Save' buttons. The 'Advanced' tab is also visible.

2.2 Requirements in Detail (Ambari)

Block Replication 설정 확인

The screenshot shows the Ambari web interface for managing HDFS configurations. The URL in the address bar is `namenode.hadoop.com:8080/#/main/services/HDFS/configs`. The top navigation bar includes tabs for Dashboard, Services, Hosts, Alerts, Admin, and a user dropdown for 'admin'. A banner at the top indicates 'Restart Required: 1 Component on 1 Host' with a 'Restart' button. On the left, a sidebar lists various services: HDFS (selected), YARN, MapReduce2, Tez, Hive, Pig, Sqoop, ZooKeeper, Ambari Metrics, SmartSense, Spark, and Zeppelin. The main content area displays the 'Configs' tab for HDFS, showing a list of configuration versions (V8, V7, V6, V5, V4, V3) with details like author ('admin'), date ('4 days ago' or '5 days ago'), and HDP version ('HDP-2.6'). A message at the bottom states 'admin authored on Fri, Nov 16, 2018 19:29'. At the bottom of the page, there are 'Settings' and 'Advanced' tabs, with 'Advanced' being highlighted with a red border.

2.2 Requirements in Detail (Ambari)

Block Replication 설정 확인

The screenshot shows the Ambari web interface for managing a cluster named 'Cluster01'. The left sidebar is collapsed, showing a list of services: HDFS, YARN, MapReduce2, Tez, Hive, Pig, Sqoop, ZooKeeper, Ambari Metrics, SmartSense, Spark, Zeppelin, Notebook, and Slider. The main content area displays the 'Configs' tab for the HDFS service. A yellow banner at the top indicates 'Restart Required: 1 Component on 1 Host'. Below this, the 'Manage Config Groups' section shows a list of configurations (V8, V7, V6, V5, V4, V3) with their details: author (admin), date (4 days ago or 5 days ago), and component (HDP-2.6). The 'V8' configuration is selected. At the bottom of this section, there are 'Discard' and 'Save' buttons. The 'General' settings section contains a field labeled 'Block replication' with the value '2', which is highlighted by a red rectangular box. There are also icons for lock, add, and copy.

2.2 Requirements in Detail (Non-Ambari)

Block Replication 및 size 설정 확인

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>/home/hadoop/data/dfs/namenode</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>/home/hadoop/data/dfs/datanode</value>
    </property>
    <property>
        <name>dfs.journalnode.edits.dir</name>
        <value>/home/hadoop/data/dfs/journalnode</value>
    </property>
    <property>
        <name>dfs.nameservices</name>
        <value>hadoop-cluster</value>
    </property>
    <property>
        <name>dfs.ha.namenodes.hadoop-cluster</name>
        <value>nn1,nn2</value>
    </property>
    <property>
        <name>dfs.namenode.rpc-address.hadoop-cluster.nn1</name>
        <value>jmj:8020</value>
    </property>
    <property>
        <name>dfs.namenode.rpc-address.hadoop-cluster.nn2</name>
        <value>sdh:8020</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hadoop-cluster.nn1</name>
        <value>jmj:50070</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hadoop-cluster.nn2</name>
        <value>sdh:50090</value>
    </property>
```

```
<property>
    <name>dfs.namenode.shared.edits.dir</name>
    <value>qjournal://jmj:8485;sdh:8485;khk:8485/hadoop-cluster</value>
</property>
<property>
    <name>dfs.client.failover.proxy.provider.hadoop-cluster</name>
    <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>
<property>
    <name>dfs.ha.fencing.methods</name>
    <value>sshfence</value>
</property>
<property>
    <name>dfs.ha.fencing.ssh.private-key-files</name>
    <value>/home/hadoop/.ssh/id_rsa</value>
</property>
<property>
    <name>dfs.ha.automatic-failover.enabled</name>
    <value>true</value>
</property>
<property>
    <name>dfs.block.size</name>
    <value>33554432</value>
</property>
<property>
    <name>dfs.hosts.exclude</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/exclude_list</value>
</property>
<property>
    <name>mapred.hosts.exclude</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/exclude_list</value>
</property>
<property>
    <name>dfs.hosts</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/include_list</value>
</property>
</configuration>
```

03 | Service Usage Process

BY DELIGHT

HDP 관리 페이지 접속 방법



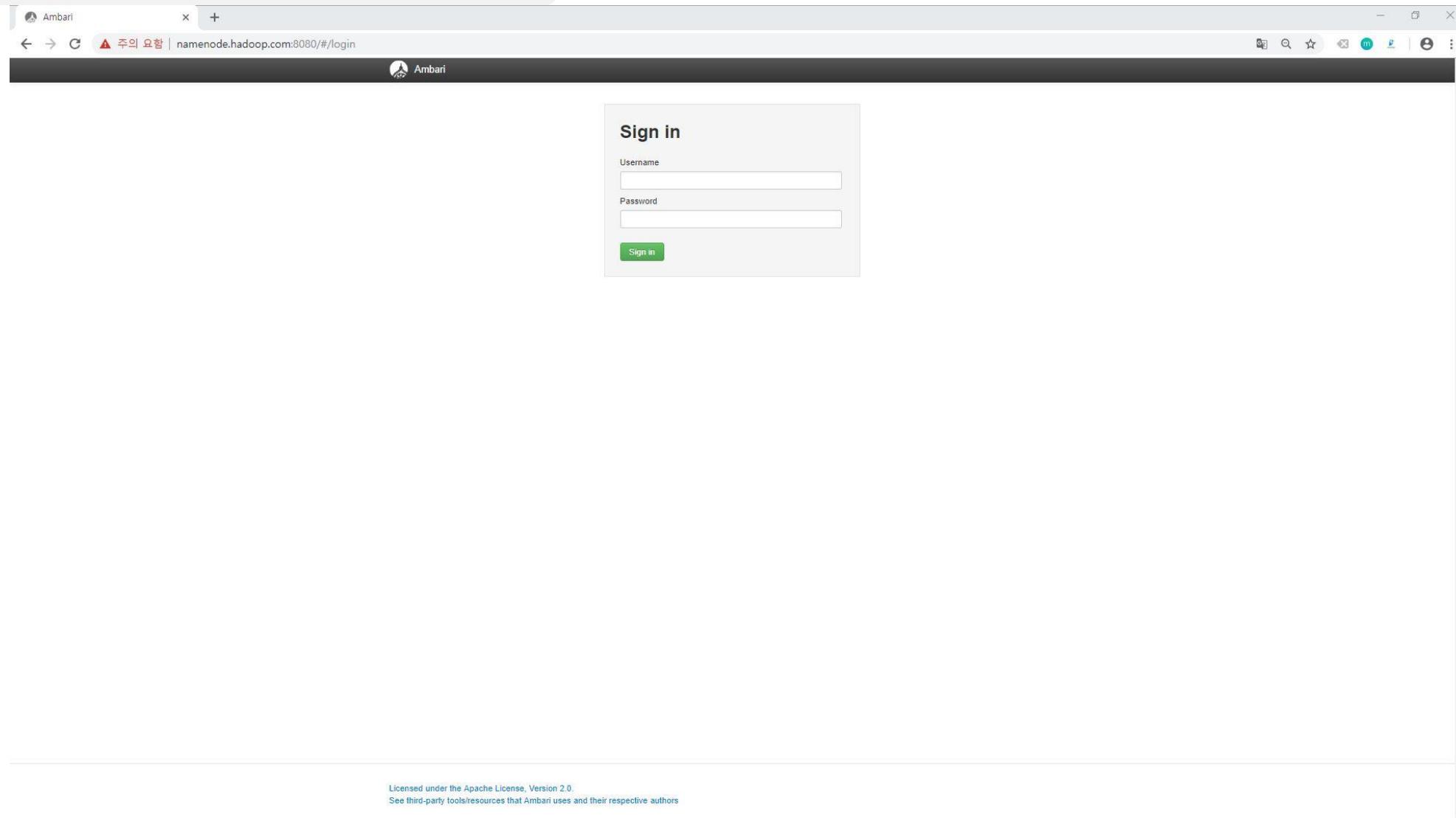
3.1 HDP 관리 페이지 접속 방법

Ambari

- 1 1. Namenode의 가상 머신에서 다음 과정 진행.
- 2
- 3 [root@host명 ~]# systemctl stop firewalld (방화벽 해제)
- 4 [root@host명 ~]# ambari-server start (ambari server 시작)
- 5 [root@host명 ~]# abmari-agent start (ambari agent 시작)
- 6
- 7 2. 웹 브라우저의 URL 주소 입력부분에 HDP 관리자 페이지를 실행할 URL 입력.
- 8 http://'namenode'의 host명 또는 ip주소':8080
- 9
- 10 3. HDP 관리자 페이지가 실행되면 Username : admin / Password : admin 입력

3.1 HDP 관리 페이지 접속 방법

Ambri



3.1 HDP 관리 페이지 접속 방법

Ambari

Ambari - Cluster01

주의 요함 | namenode.hadoop.com:8080/#main/dashboard/metrics

Ambari Cluster01 0 ops 1 alert

Dashboard Services Hosts Alerts Admin admin

Metrics Heatmaps Config History

Metric Actions Last 1 hour

HDFS Disk Usage 10% DataNodes Live 3/3 HDFS Links NameNode Secondary NameNode 3 DataNodes Memory Usage 4.6 GB Network Usage 19.5 KB

CPU Usage 100% Cluster Load 4/4 NameNode Heap 4% NameNode RPC 7.25 ms NameNode CPU WIO 0.1%

NameNode Uptime 3.1 hr ResourceManager Heap 18% ResourceManager Uptime 4.4 hr YARN Memory 0% NodeManagers Live 3/3

YARN Links ResourceManager 3 NodeManagers

Licensed under the Apache License, Version 2.0.
See third-party tools/resources that Ambari uses and their respective authors

The screenshot shows the Ambari Metrics dashboard for Cluster01. The top navigation bar includes links for Dashboard, Services, Hosts, Alerts, Admin, and a user dropdown for 'admin'. Below the navigation is a sub-menu with tabs for Metrics, Heatmaps, and Config History, along with Metric Actions and a time range selector for the last hour. The main content area displays various metrics in cards:

- HDFS Disk Usage:** 10% (Circular gauge)
- DataNodes Live:** 3/3 (Text and green indicator)
- HDFS Links:** NameNode, Secondary NameNode, 3 DataNodes (Text and 'More...' button)
- Memory Usage:** 4.6 GB (Bar chart)
- Network Usage:** 19.5 KB (Line chart)
- CPU Usage:** 100% (Stacked bar chart)
- Cluster Load:** 4/4 (Line chart)
- NameNode Heap:** 4% (Circular gauge)
- NameNode RPC:** 7.25 ms (Text)
- NameNode CPU WIO:** 0.1% (Circular gauge)
- NameNode Uptime:** 3.1 hr (Text)
- ResourceManager Heap:** 18% (Circular gauge)
- ResourceManager Uptime:** 4.4 hr (Text)
- YARN Memory:** 0% (Circular gauge)
- NodeManagers Live:** 3/3 (Text)
- YARN Links:** ResourceManager, 3 NodeManagers (Text and 'More...' button)

At the bottom, a note states: "Licensed under the Apache License, Version 2.0. See third-party tools/resources that Ambari uses and their respective authors".

3.1 HDP 관리 페이지 접속 방법

Non Ambari

1. **zookeeper**가 설치된 각 서버에서 다음 과정 진행

```
[zookeeper@jmj ~]$ source .bashrc  
[zookeeper@jmj ~]$ zkServer.sh .start  
[zookeeper@jmj ~]$ zkServer.sh .status
```

Mode ghkrdls : leader(1), follower(2)

2. **jmj** 호스트에서 **zookeeper format**

```
[hadoop@jmj ~]$ hdfs zkfc -formatZK
```

3. 각각의 **journalnode** 실행

```
[hadoop@jmj ~]$ hadoop-daemon.sh start journalnode
```

4. **jmj** 호스트에서 **namenode** 포맷

```
[hadoop@jmj ~]$ hdfs namenode -format
```

5. **jmj** 호스트에서 **namenode** 및 **zkfc** 실행

```
[hadoop@jmj ~]$ hadoop-daemon.sh start namenode  
[hadoop@jmj ~]$ hadoop-daemon.sh start zkfc
```

3.1 HDP 관리 페이지 접속 방법

Non Ambari

6. jmj 호스트에서 datanodes 실행

```
[hadoop@jmj ~]$ hadoop-daemons.sh start datanode
```

7. sdh 호스트에서 standbynamenode 및 zkfc 실행

```
[hadoop@jmj ~]$ hdfs namenode -bootatarpStandby  
[hadoop@jmj ~]$ hadoop-daemon.sh start namenode  
[hadoop@jmj ~]$ hadoop-daemons.sh start zkfc
```

8. 하둡 분산 정상 설치 여부 확인 (wordcount실행)

```
[hadoop@jmj ~]$ hdfs dfs -put /home/hadoop/hadoop2/etc/hadoop/hadoop-env.sh /user/hadoop/conf  
[hadoop@jmj ~]$ yarn jar /home/hadoop/hadoop2/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar wordcount conf output
```

9. jmj 호스트에서 yarn-cluster 실행

```
[hadoop@jmj ~]$ start-yarn.sh
```

10. jmj 호스트에서 historyserver 실행

```
[hadoop@jmj ~]$ mr-jobhistory-daemon.sh start historyserver
```

11. jmj 호스트에서 proxyserver 실행

```
[hadoop@jmj ~]$ yarn-demon.sh start proxyserver
```

3.1 HDP 관리 페이지 접속 방법

Non Ambari

```
[hadoop@jmj sbin]$ jps  
3504 NameNode  
5106 NodeManager  
4996 ResourceManager  
4203 JobHistoryServer  
3595 DFSZKFailoverController  
3389 JournalNode  
3725 DataNode  
5327 Jps
```

```
[hadoop@khk hadoop]$ jps  
8385 NodeManager  
7594 JournalNode  
8429 Jps  
7677 DataNode
```

```
[hadoop@sdh ~]$ jps  
2085 JournalNode  
2168 DataNode  
2393 DFSZKFailoverController  
2300 NameNode  
3983 Jps
```

```
[hadoop@lkh ~]$ jps  
2900 Jps  
2842 NodeManager  
1947 DataNode
```

3.1 HDP 관리 페이지 접속 방법

Non Ambari

| Hadoop | Overview | Datanodes | Snapshot | Startup Progress | Utilities ▾ |
|--------|----------|-----------|----------|------------------|-------------|
|--------|----------|-----------|----------|------------------|-------------|

Overview 'jmj:8020' (active)

| | |
|-----------------------|---|
| Started: | Tue Nov 20 16:11:20 KST 2018 |
| Version: | 2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1 |
| Compiled: | 2014-11-13T21:10Z by jenkins from (detached from e349649) |
| Cluster ID: | CID-63c9e580-5242-4382-a311-803a5b477029 |
| Block Pool ID: | BP-1891888096-192.168.103.143-1542697857159 |

3.1 HDP 관리 페이지 접속 방법

Non Ambari

Datanode Information

In operation

| Node | Last contact | Admin State | Capacity | Used | Non DFS Used | Remaining | Blocks | Block pool used | Failed Volumes | Version |
|-----------------------------|--------------|----------------|----------|-----------|--------------|-----------|--------|-------------------|----------------|---------|
| jmj (192.168.103.143:50010) | 1 | Decommissioned | 31.18 GB | 24 KB | 6.06 GB | 25.12 GB | 0 | 24 KB (0%) | 0 | 2.6.0 |
| sdh (192.168.103.142:50010) | 1 | In Service | 27.44 GB | 683.03 MB | 3.7 GB | 23.07 GB | 44 | 683.03 MB (2.43%) | 0 | 2.6.0 |
| lkh (192.168.103.144:50010) | 0 | In Service | 31.18 GB | 747.88 MB | 2.87 GB | 27.57 GB | 50 | 747.88 MB (2.34%) | 0 | 2.6.0 |
| khk (192.168.103.141:50010) | 1 | In Service | 31.18 GB | 834.75 MB | 3.41 GB | 26.95 GB | 50 | 834.75 MB (2.61%) | 0 | 2.6.0 |

3.1 HDP 관리 페이지 접속 방법

Non Ambari

Browse Directory

| /user/hadoop | | | | | | | Go! |
|--------------|--------|------------|------|-------------|------------|--------|-----|
| Permission | Owner | Group | Size | Replication | Block Size | Name | |
| drwxr-xr-x | hadoop | supergroup | 0 B | 0 | 0 B | conf | |
| drwxr-xr-x | hadoop | supergroup | 0 B | 0 | 0 B | output | |

Browse Directory

| /user/hadoop/output | | | | | | | Go! |
|---------------------|--------|------------|---------|-------------|------------|--------------|-----|
| Permission | Owner | Group | Size | Replication | Block Size | Name | |
| -rw-r--r-- | hadoop | supergroup | 0 B | 2 | 32 MB | _SUCCESS | |
| -rw-r--r-- | hadoop | supergroup | 1.46 KB | 2 | 32 MB | part-r-00000 | |

```
[hadoop@client hadoop2]$ hdfs dfs -cat /user/hadoop/output/part-r-00000
10      1
Apart   1
Auto-Save    1
Bookmark    1
Details    1
Disclaimer: 1
For       1
In       2
```

3.1 HDP 관리 페이지 접속 방법

Non Ambari

주의 요함 | 192.168.103.143:8088/cluster

Apps Data_Science SW AWS PA Chart JDK_API Visual Studio Code Live Server MobaXterm Cloudera FileZilla DA과제 Apache jsoup CSS encore

Logged in as: dr.who

 All Applications

| Cluster Metrics | | | | | | | | | | | | | | | | | |
|---|----------------|--------------|--------------|----------------|--------------------|-------------|--------------|-----------------|-------------|--------------|-----------------|--------------|----------------------|------------|-----------------|----------------|--|
| About Nodes Applications | Apps Submitted | Apps Pending | Apps Running | Apps Completed | Containers Running | Memory Used | Memory Total | Memory Reserved | Vcores Used | Vcores Total | Vcores Reserved | Active Nodes | Decommissioned Nodes | Lost Nodes | Unhealthy Nodes | Rebooted Nodes | |
| NEW NEW_SAVING SUBMITTED ACCEPTED RUNNING FINISHED FAILED KILLED | 14 | 14 | 0 | 0 | 0 | 0 B | 0 B | 0 B | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Show 20 entries Search:

| ID | User | Name | Application Type | Queue | StartTime | FinishTime | State | FinalStatus | Progress | Tracking UI |
|--------------------------------|--------|------------|------------------|---------|-------------------------------|------------|----------|-------------|----------|-------------|
| application_1542698264452_0015 | hadoop | word count | MAPREDUCE | default | Tue, 20 Nov 2018 09:55:39 GMT | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |
| application_1542698264452_0014 | hadoop | word count | MAPREDUCE | default | Tue, 20 Nov 2018 09:45:13 GMT | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |
| application_1542698264452_0013 | hadoop | word count | MAPREDUCE | default | Tue, 20 Nov 2018 09:43:59 GMT | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |
| application_1542698264452_0012 | hadoop | word count | MAPREDUCE | default | Tue, 20 Nov 2018 08:52:46 GMT | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |
| application_1542698264452_0011 | hadoop | word count | MAPREDUCE | default | Tue, 20 Nov 2018 08:32:15 GMT | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |
| application_1542698264452_0010 | hadoop | Spark Pi | SPARK | default | Tue, 20 Nov 2018 08:21:05 GMT | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |
| application_1542698264452_0009 | hadoop | Spark Pi | SPARK | default | Tue, 20 Nov 2018 08:18:05 GMT | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |
| application_1542698264452_0008 | hadoop | Spark Pi | SPARK | default | Tue, 20 Nov 2018 08:14:55 | N/A | ACCEPTED | UNDEFINED | | UNASSIGNED |

Scheduler Tools



HIVE

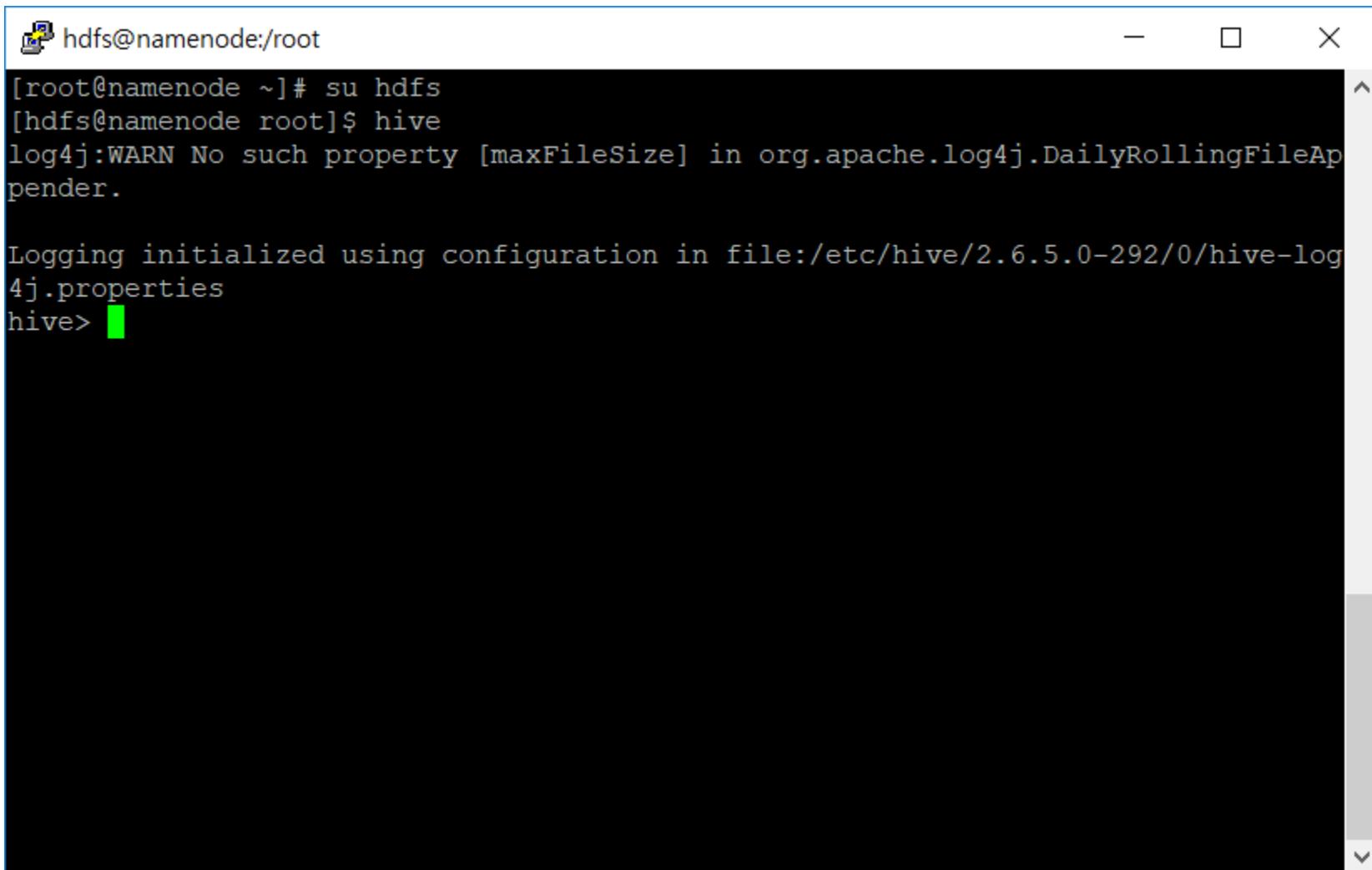
3.2 Hive 접속 방법

Ambari

```
1 1. Namenode의 가상 머신에서 root 계정에서 hdfs 계정으로 접속.
2
3 [root@host명 ~]# su hdfs
4 [hdfs@host명 root]$
5
6 2. hdfs 계정의 /user/hive 디렉토리의 하위에 연습용으로 사용할 tests 디렉토리 생성.
7 [hdfs@host명 root]$ hdfs dfs -mkdir /user/hive/tests
8
9 3. hive 접속 (실행되는 시간 다소 소요)
10 [hdfs@host명 root]$ hive
11 hive>
12
13 4. hdfs 계정에 새로 생성한 디렉토리인 /user/hive/tests를 확인하기 위해
14 웹 브라우저에 URL 주소 입력 부분에 다음의 내용을 입력.
15 http://'namenode의 host명 또는 ip주소':50070
16
17 50070은 hdfs의 기본 namenode의 포트번호.
18
19 5. 웹 UI 화면이 출력된 후 화면의 상단 메뉴 중 utilites에
20 Brows the file system을 선택하면 hdfs 계정의 디렉토리 확인 가능.
21
22 6. 화면에 나타난 디렉토리를 선택하면서 이동.
23 user 디렉토리를 선택하고 hive 디렉토리를 선택하면 tests 디렉토리 확인.
```

3.2 Hive 접속 방법

Ambari



hdfs@namenode:/root

```
[root@namenode ~]# su hdfs
[hdfs@namenode root]$ hive
log4j:WARN No such property [maxFileSize] in org.apache.log4j.DailyRollingFileAppender.

Logging initialized using configuration in file:/etc/hive/2.6.5.0-292/0/hive-log4j.properties
hive>
```

3.2 Hive 접속 방법

Ambari

The screenshot shows the Ambari web interface with a green header bar. On the left, there's a green arrow pointing right with the word "Ambari" next to it. The main content area has a title bar with "Namenode information" and a close button. Below the title bar is a navigation bar with back, forward, and search icons, followed by the URL "namenode.hadoop.com:50070/dfshealth.html#tab-overview". The main content area has a green header bar with tabs: "Hadoop" (selected), "Overview" (highlighted in green), "Datanodes", "Datanode Volume Failures", "Snapshot", "Startup Progress", and "Utilities" (highlighted with a red box). The main content below the header is titled "Overview 'namenode.hadoop.com:8020' (active)".

Overview 'namenode.hadoop.com:8020' (active)

3.2 Hive 접속 방법

Ambari

Namenode information x +

← → ⌂ ⓘ 주의 요함 | namenode.hadoop.com:50070/dfshealth.html#tab-overview

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities ▾

Browse the file system
Logs

Overview 'namenode.hadoop.com:8020' (active)

3.2 Hive 접속 방법

Ambari

Browse Directory

| Browse Directory | | | | | | | | Go! |
|------------------|--------|--------|------|---------------------------|-------------|------------|----------------|-----|
| Permission | Owner | Group | Size | Last Modified | Replication | Block Size | Name | |
| drwxrwxrwx | yarn | hadoop | 0 B | 2018. 11. 15. 오후 5:18:03 | 0 | 0 B | app-logs | |
| drwxr-xr-x | hdfs | hdfs | 0 B | 2018. 11. 15. 오후 4:30:21 | 0 | 0 B | apps | |
| drwxr-xr-x | yarn | hadoop | 0 B | 2018. 11. 15. 오전 9:37:51 | 0 | 0 B | ats | |
| drwxr-xr-x | hdfs | hdfs | 0 B | 2018. 11. 15. 오전 9:37:44 | 0 | 0 B | hdp | |
| drwxr-xr-x | mapred | hdfs | 0 B | 2018. 11. 15. 오전 9:37:40 | 0 | 0 B | mapred | |
| drwxrwxrwx | mapred | hadoop | 0 B | 2018. 11. 15. 오전 9:38:06 | 0 | 0 B | mr-history | |
| drwxrwxrwx | spark | hadoop | 0 B | 2018. 11. 20. 오후 5:41:55 | 0 | 0 B | spark-history | |
| drwxrwxrwx | spark | hadoop | 0 B | 2018. 11. 16. 오후 7:23:56 | 0 | 0 B | spark2-history | |
| drwxrwxrwx | hdfs | hdfs | 0 B | 2018. 11. 15. 오후 12:56:14 | 0 | 0 B | tmp | |
| drwxr-xr-x | hdfs | hdfs | 0 B | 2018. 11. 16. 오후 2:00:09 | 0 | 0 B | user | |

3.2 Hive 접속 방법

Ambari

Browse Directory

| Browse Directory | | | | | | | |
|------------------|-----------|-------|------|---------------------------|-------------|------------|-----------|
| Permission | Owner | Group | Size | Last Modified | Replication | Block Size | Name |
| drwxr-xr-x | hdfs | hdfs | 0 B | 2018. 11. 16. 오후 2:00:09 | 0 | 0 B | admin |
| drwxrwx--- | ambari-qa | hdfs | 0 B | 2018. 11. 15. 오후 12:54:03 | 0 | 0 B | ambari-qa |
| drwxr-xr-x | hcat | hdfs | 0 B | 2018. 11. 15. 오후 12:50:39 | 0 | 0 B | hcat |
| drwxr-xr-x | hdfs | hdfs | 0 B | 2018. 11. 16. 오후 12:34:52 | 0 | 0 B | hdfs |
| drwxr-xr-x | hive | hdfs | 0 B | 2018. 11. 16. 오후 4:07:19 | 0 | 0 B | hive |
| drwxrwxr-x | spark | spark | 0 B | 2018. 11. 19. 오전 11:13:56 | 0 | 0 B | spark |
| drwxr-xr-x | zeppelin | hdfs | 0 B | 2018. 11. 19. 오전 10:06:19 | 0 | 0 B | zeppelin |

3.2 Hive 접속 방법

Ambari

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities ▾

Browse Directory

/user/hive Go!

| Permission | Owner | Group | Size | Last Modified | Replication | Block Size | Name |
|------------|-------|-------|------|--------------------------|-------------|------------|-------|
| drwxr-xr-x | hdfs | hdfs | 0 B | 2018. 11. 16. 오후 5:52:56 | 0 | 0 B | tests |

Spark History Server 접속 방법



3.3 Spark History Server 접속 방법

Ambari

Ambari - Cluster01 0 ops 1 alert

주의 요함 | namenode.hadoop.com:8080/#/main/dashboard/metrics

Dashboard Services Hosts Alerts Admin admin

HDFS YARN MapReduce2 Tez Hive Pig Sqoop ZooKeeper Ambari Metrics SmartSense **Spark** Zeppelin Notebook Slider Actions ▾

Metrics Heatmaps Config History Metric Actions ▾ Last 1 hour ▾

HDFS Disk Usage 10% DataNodes Live 3/3 HDFS Links NameNode Secondary NameNode 3 DataNodes More... Memory Usage 4.6 GB Network Usage 19.5 KB

CPU Usage 100% 50% Cluster Load 4 2 NameNode Heap 4% NameNode RPC 7.25 ms NameNode CPU WIO 0.1%

NameNode Uptime 3.1 hr ResourceManager Heap 18% ResourceManager Uptime 4.4 hr YARN Memory 0% NodeManagers Live 3/3

YARN Links ResourceManager 3 NodeManagers More... ▾

Licensed under the Apache License, Version 2.0.
See third-party tools/resources that Ambari uses and their respective authors

The screenshot shows the Ambari Metrics dashboard for Cluster01. The left sidebar lists services: HDFS, YARN, MapReduce2, Tez, Hive, Pig, Sqoop, ZooKeeper, Ambari Metrics, SmartSense, **Spark**, Zeppelin Notebook, and Slider. The 'Spark' service is highlighted with a red box. The main area displays various metrics in cards: HDFS Disk Usage (10%), DataNodes Live (3/3), HDFS Links (NameNode, Secondary NameNode, 3 DataNodes), Memory Usage (4.6 GB), Network Usage (19.5 KB), CPU Usage (100%, 50%), Cluster Load (4, 2), NameNode Heap (4%), NameNode RPC (7.25 ms), NameNode CPU WIO (0.1%), NameNode Uptime (3.1 hr), ResourceManager Heap (18%), ResourceManager Uptime (4.4 hr), YARN Memory (0%), and NodeManagers Live (3/3). A 'YARN Links' section at the bottom shows ResourceManager and 3 NodeManagers.

3.3 Spark History Server 접속 방법

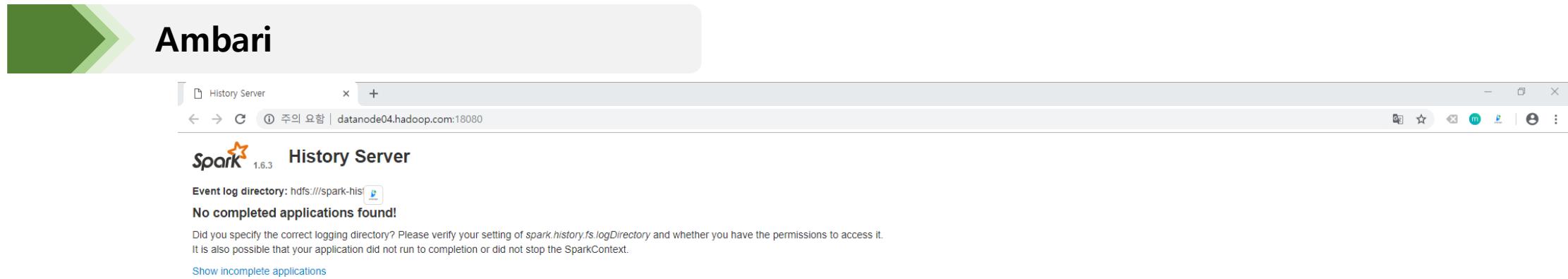
Ambari

The screenshot shows the Ambari web interface for Cluster01. The left sidebar lists various services: HDFS, YARN, MapReduce2, Tez, Hive, Pig, Sqoop, ZooKeeper, Ambari Metrics, SmartSense, Spark (which is selected), Zeppelin, Notebook, and Slider. The main content area has tabs for Summary and Configs, with Summary selected. A 'Quick Links' dropdown menu is open, and the 'Spark History Server UI' option is highlighted with a red box. Below the dropdown, there are service status cards: Spark History Server (Started, No alerts), Livy Server (0/0 Live), Spark Thrift Server (0/0 Live), and Spark Clients (5 Installed). The top navigation bar shows 'Dashboard', 'Services', 'Hosts', 'Alerts', 'Admin', and a user dropdown for 'admin'. The URL in the browser is 'namenode.hadoop.com:8080/#/main/services/SPARK/summary'.

Licensed under the Apache License, Version 2.0.
See third-party tools/resources that Ambari uses and their respective authors

datanode04.hadoop.com:18080

3.3 Spark History Server 접속 방법



3.3 Spark History Server 접속 방법

Non Ambari

```
1. jmj 호스트(master)에서 master, slaves 실행  
[hadoop@jmj sbin]$ start-master.sh  
[hadoop@jmj sbin]$ start-slaves.sh  
2. spark test  
[hadoop@lkh sbin]$ spark-submit --master spark://jmj:7077 --class org.apache.spark.examples.SparkPi /home/hadoop/spark2/examples/jars/spark-examples*.jar 10  
3. spark history-server 실행  
[hadoop@jmj ~]$ hdfs dfs -mkdir /spark-logs  
[hadoop@jmj ~]$ hdfs dfs -chmod 777 /spark-logs  
[hadoop@jmj conf]$ start-history-server.sh
```

```
[hadoop@jmj sbin]$ jps  
3504 NameNode  
5609 Master  
4996 ResourceManager  
6679 Jps  
4203 JobHistoryServer  
3595 DFSZKFailoverController  
3389 JournalNode  
3725 DataNode  
^C[hadoop@jmj sbin]$
```

```
8724 Worker  
8806 Jps  
7594 JournalNode  
7677 DataNode
```

```
[hadoop@sdh ~]$ jps  
2085 JournalNode  
2168 DataNode  
4504 Worker  
2393 DFSZKFailoverController  
4556 Jps  
2300 NameNode
```

```
[hadoop@lkh ~]$ jps  
3207 Worker  
3257 Jps  
1947 DataNode
```

3.3 Spark History Server 접속 방법

Non Ambari

주의 요함 | 192.168.103.143:50070/explorer.html#/user/hadoop/.sparkStaging

Science G SW AWS PA Chart JDK_API Visual Studio Code Live Server MobaXterm Cloudera FileZilla DA과제 Apache jsoup CSS

Hadoop Overview Datanodes Snapshot Startup Progress Utilities ▾

Browse Directory

/user/hadoop/.sparkStaging

| Permission | Owner | Group | Size | Replication | Block Size | Name |
|------------|--------|------------|------|-------------|------------|--|
| drwx----- | hadoop | supergroup | 0 B | 0 | 0 B | application_1542698264452_0005 |
| drwx----- | hadoop | supergroup | 0 B | 0 | 0 B | application_1542698264452_0006 |
| drwx----- | hadoop | supergroup | 0 B | 0 | 0 B | application_1542698264452_0007 |
| drwx----- | hadoop | supergroup | 0 B | 0 | 0 B | application_1542698264452_0008 |
| drwx----- | hadoop | supergroup | 0 B | 0 | 0 B | application_1542698264452_0009 |
| drwx----- | hadoop | supergroup | 0 B | 0 | 0 B | application_1542698264452_0010 |

3.3 Spark History Server 접속 방법

Non Ambari

 2.1.3 Spark Master at spark://jmj:7077

URL: spark://jmj:7077

REST URL: spark://jmj:6066 (*cluster mode*)

Alive Workers: 3

Cores in use: 6 Total, 0 Used

Memory in use: 23.6 GB Total, 0.0 B Used

Applications: 0 [Running](#), 0 [Completed](#)

Drivers: 0 Running, 0 Completed

Status: ALIVE

Workers

| Worker Id | Address | State | Cores | Memory |
|---|-----------------------|-------|------------|---------------------|
| worker-20181116182131-192.168.103.141-41356 | 192.168.103.141:41356 | ALIVE | 2 (0 Used) | 9.4 GB (0.0 B Used) |
| worker-20181120165443-192.168.103.142-35904 | 192.168.103.142:35904 | ALIVE | 2 (0 Used) | 9.4 GB (0.0 B Used) |
| worker-20181120165744-192.168.103.144-38792 | 192.168.103.144:38792 | ALIVE | 2 (0 Used) | 4.8 GB (0.0 B Used) |

Running Applications

| Application ID | Name | Cores | Memory per Node | Submitted Time | User | State | Duration |
|----------------|------|-------|-----------------|----------------|------|-------|----------|
| | | | | | | | |

Completed Applications

| Application ID | Name | Cores | Memory per Node | Submitted Time | User | State | Duration |
|----------------|------|-------|-----------------|----------------|------|-------|----------|
| | | | | | | | |

3.3 Spark History Server 접속 방법

Non Ambari

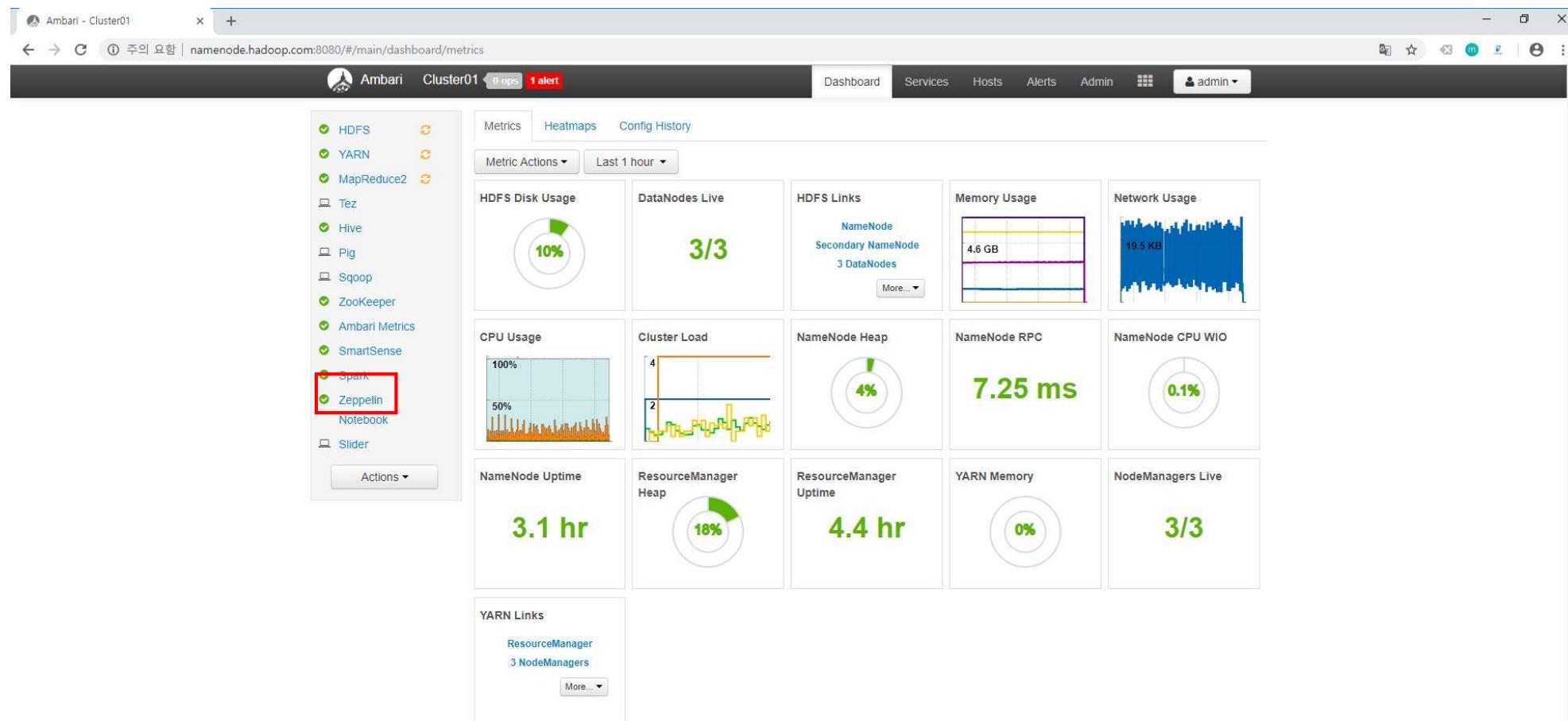
```
18/11/22 16:46:27 INFO TaskSetManager: Finished task 2.0 in stage 0.0 (TID 2) in  
1347 ms on 192.168.103.144 (executor 1) (9/10)  
18/11/22 16:46:27 INFO TaskSetManager: Finished task 3.0 in stage 0.0 (TID 3) in  
1315 ms on 192.168.103.144 (executor 1) (10/10)  
18/11/22 16:46:27 INFO DAGScheduler: ResultStage 0 (reduce at SparkPi.scala:38) f  
inished in 3.556 s  
18/11/22 16:46:27 INFO TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks have a  
ll completed, from pool  
18/11/22 16:46:27 INFO DAGScheduler: Job 0 finished: reduce at SparkPi.scala:38,  
task 1.614561 s  
Pi is roughly 3.144335144335144  
18/11/22 16:46:27 INFO SparkUI: Stopped Spark web UI at http://192.168.103.144:40  
40  
18/11/22 16:46:27 INFO StandaloneSchedulerBackend: Shutting down all executors  
18/11/22 16:46:27 INFO CoarseGrainedSchedulerBackend$DriverEndpoint: Asking each  
executor to shut down  
18/11/22 16:46:28 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndp  
oint stopped!  
18/11/22 16:46:28 INFO MemoryStore: MemoryStore cleared  
18/11/22 16:46:28 INFO BlockManager: BlockManager stopped
```

Zeppelin Notebook 접속 방법

Apache Zeppelin

3.4 Zeppelin Notebook 접속 방법

Ambri



3.4 Zeppelin Notebook 접속 방법

Ambari

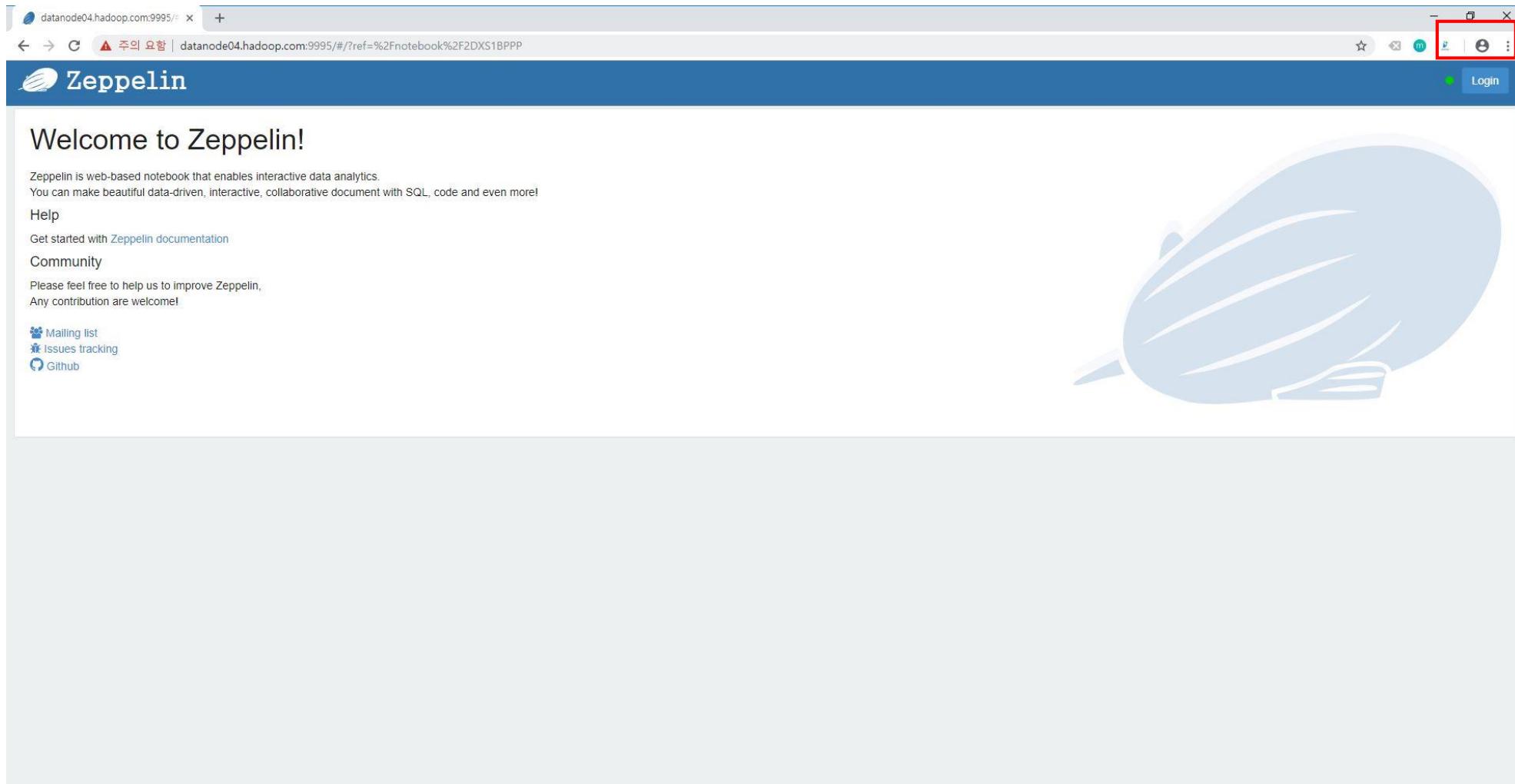
The screenshot shows the Ambari web interface for Cluster01. The left sidebar lists services: HDFS, YARN, MapReduce2, Tez, Hive, Pig, Sqoop, ZooKeeper, Ambari Metrics, SmartSense, Spark, Zeppelin Notebook, and Slider. The Zeppelin Notebook service is highlighted with a red box around its entry in the sidebar. On the right, the main dashboard shows a summary for the Zeppelin Notebook service, indicating it is Started and has No alerts. A red box highlights the "Zeppelin UI" link under the "Quick Links" section. The top navigation bar shows 0 ops and 1 alert.

Licensed under the Apache License, Version 2.0.
See third-party tools/resources that Ambari uses and their respective authors

datanode04.hadoop.com:9995

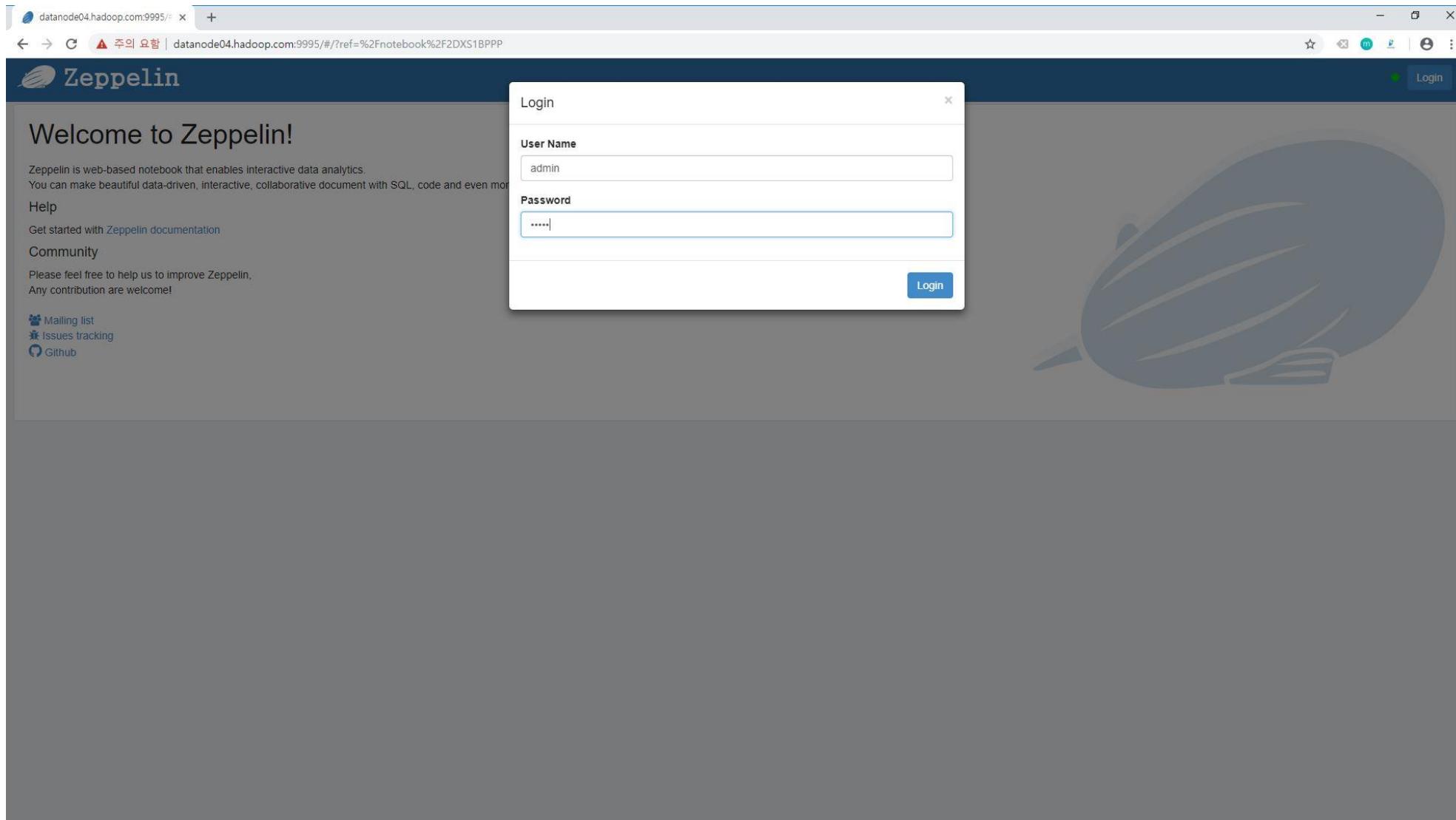
3.4 Spark History Server 접속 방법

Ambari



3.4 Spark History Server 접속 방법

Ambari



3.4 Spark History Server 접속 방법

Ambari

The screenshot shows a web browser window titled "datanode04.hadoop.com:9995/#/" with the Zeppelin logo. The main content is the "Welcome to Zeppelin!" page. It includes a brief introduction: "Zeppelin is web-based notebook that enables interactive data analytics. You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!". Below this, there are sections for "Notebook", "Help", and "Community". The "Notebook" section has links for "Import note", "Create new note", and a "Filter" search bar. Under "Getting Started", there are links for "R (SparkR)", "Zeppelin Tutorial (Basic Features)", and "Trash". The "Help" section links to "Zeppelin documentation". The "Community" section links to "Mailing list", "Issues tracking", and "Github". A large blue graphic of a hot air balloon is on the right side of the page.

datanode04.hadoop.com:9995/#/

Zeppelin Notebook Job

Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics. You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Notebook

- Import note
- Create new note
- Filter
- Getting Started
 - R (SparkR)
 - Zeppelin Tutorial (Basic Features)
 - Trash

Help

Get started with [Zeppelin documentation](#)

Community

Please feel free to help us to improve Zeppelin, Any contribution are welcome!

- Mailing list
- Issues tracking
- Github

Search your Notes

admin

3.4 Spark History Server 접속 방법

Non Ambari

1. zeppelin 실행

```
[hadoop@jmj bin]$ zeppelin-daemon.sh start
```

```
[hadoop@jmj bin]$ jps
3504 NameNode
6609 Master
4996 ResourceManager
7544 Jps
7481 ZeppelinServer
4203 JobHistoryServer
3595 DFSZKFailoverController
3389 JournalNode
3725 DataNode
[hadoop@jmj bin]$
```

3.4 Spark History Server 접속 방법

Non Ambari

주의 요함 | 192.168.103.143:8888/#/

Apps Data_Science SW AWS PA Chart JDK_API Visual Studio Code Live Server MobaXterm Cloudera FileZilla DA과제 Apache jsoup CSS encore

Zeppelin Notebook Job Search admin

Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Notebook ↗

- Import note
- Create new note

Filter

Zeppelin Tutorial

Help

Get started with [Zeppelin documentation](#)

Community

Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

Mailing list

Issues tracking

Github

04 | Demo

big
data

BY DELIGHT

05 | Errors & Corrections

BY DELIGHT

big
Data

4. Errors in Detail

Ambari

오류 발생 상황

- 모든 host의 ambari server 및 agent에 대한 설정이 완료된 후 ambari를 실행했을 때 정상적으로 실행.
- 다음 날, 모든 host를 재가동하고 ambari를 실행했을 때 서비스들이 정상적으로 실행이 되지 않는 현상 발생.

해결 방법

- Ambari-agent.ini 파일 내용 수정
- Cert-verification.cfg 파일 내용 수정

```
1 [root@host명 ~]# vi /etc/ambari-agent/conf/ambari-agent.ini
2
3     수정 1 (hostname 부분 수정)
4     hostname=localhost --> hostname=namenode의 host명 (모든 host에 적용)
5
6     수정 2 (프로토콜 설정 추가)
7     [security] 부분
8     force_https_protocol=PROTOCOL_TLSv1_2
9
10    [root@host명 ~]# vi /etc/python/cert-verification.cfg
11
12    수정 1 (verify 수정)
13    [https] 부분
14    'verify=platform_default' --> 'verify=disable'
```

4. Errors in Detail

Non Ambari

오류 발생 상황 1

- Probuf-2.5.0 설치후 configure 파일 실행 시 실행 되지 않음

해결 방법

- 컴파일러 설치 유무 확인 및 설치
 - [root@localhost protobuf-2.5.0]# rpm -qa | grep gcc (컴파일러 설치 유무 확인)
 - [root@localhost protobuf-2.5.0]# yum -y install gcc (gcc 설치)
- Development Tools 설치
 - [root@localhost protobuf-2.5.0]# yum group install "Development Tools"

오류 발생 상황 2

- Hadoop-2.6.0 설치 후 conf 디렉터리에서 설정 시 수정 불가

해결 방법

- Hadoop user에서 Hadoop-2.6.0을 재 다운로드하여 설치

오류 발생 상황 3

- JPS 실행 시 datanode가 실행 되지 않음

해결 방법

- Hadoop Home 하위의 data – dfs – datanode 디렉터리 삭제

4. Errors in Detail

Non Ambari

오류 발생 상황 4

- Spark 2.4.0 설치 시 Hadoop-2.6.0과 호환되지 않음

해결 방법

- Spark-2.1.3 재설치

오류 발생 상황 5

- JDK 7 version은 Spark-2.1.3에서 호환되지 않음

해결 방법

- JDK-8u-181 재설치

오류 발생 상황 6

- Spark history server UI (192.168.103.143:18080) 접속 불가

해결 방법

- Hadoop Namenode UI의 spark-logs 디렉터리의 권한 반영(777) 및 그룹명 변경 (supergroup -> Hadoop)
- 해결하지 못함

06 | Appendix

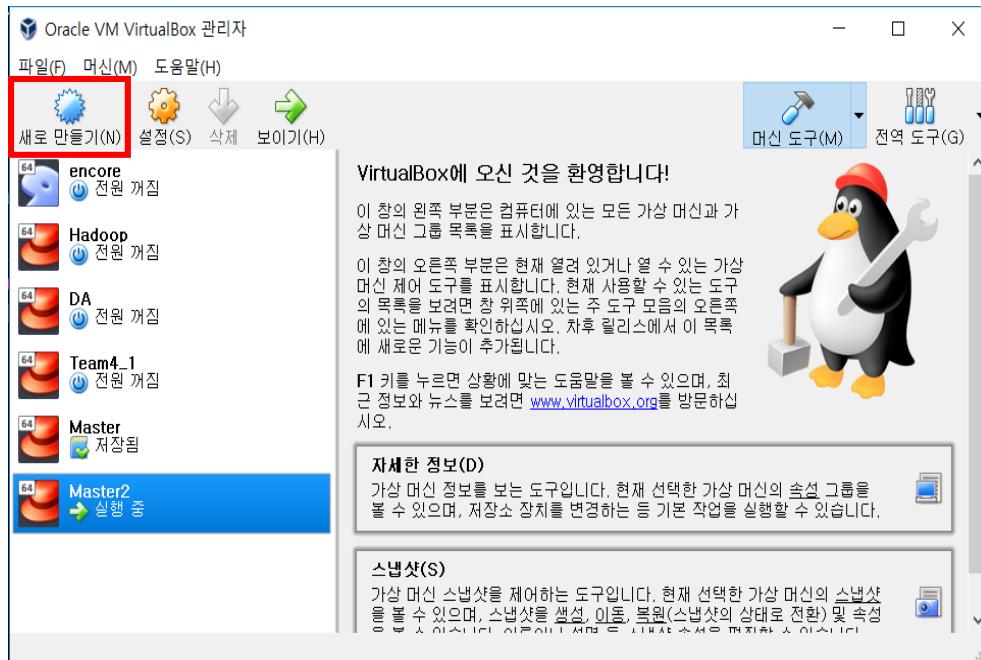
Big Data

BY DELIGHT

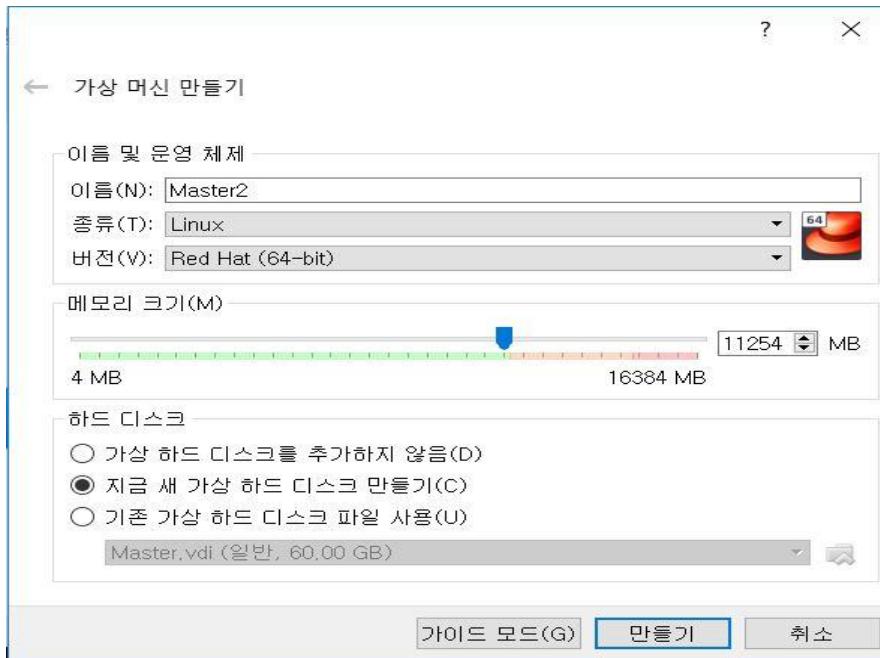


가상머신 생성

- 버추얼 박스 실행 -> 새로 만들기 선택



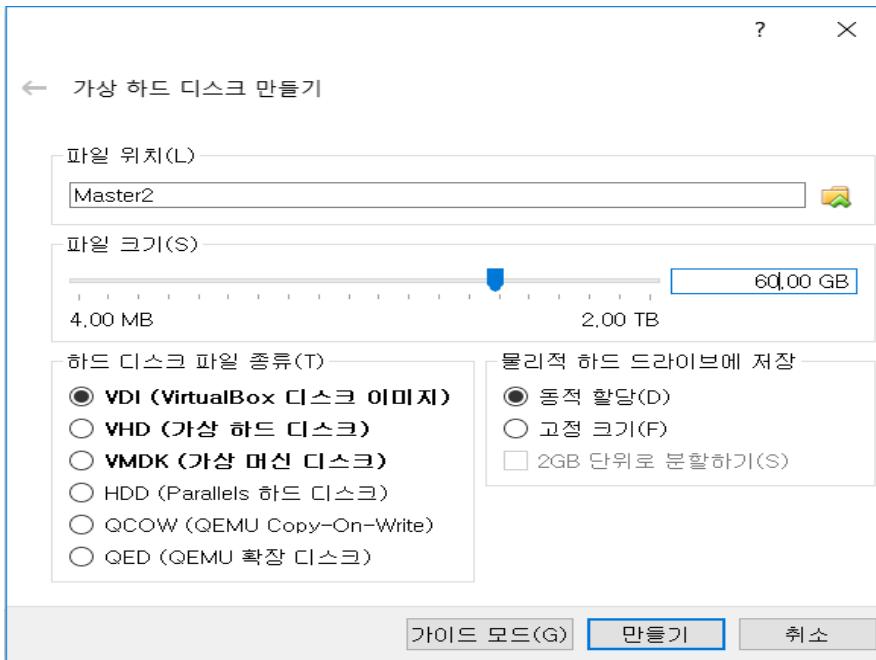
가상머신 생성



- 종류 : Linux
- 버전 : Red Hat (64-bit)
- 메모리 크기 : 녹색 부분에서 최대치
- (나머지 설정 부분은 그대로 한 후 '만들기' 선택)

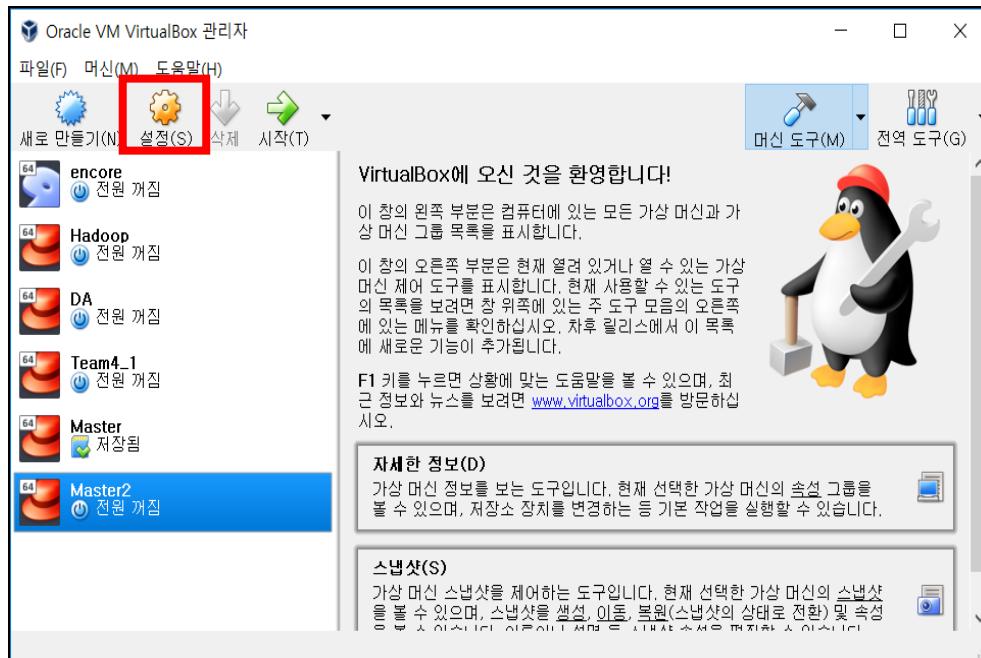
가상머신 생성

- 파일 크기 : 60 GB
- (나머지 설정 부분은 그대로 한 후 '만들기' 선택)

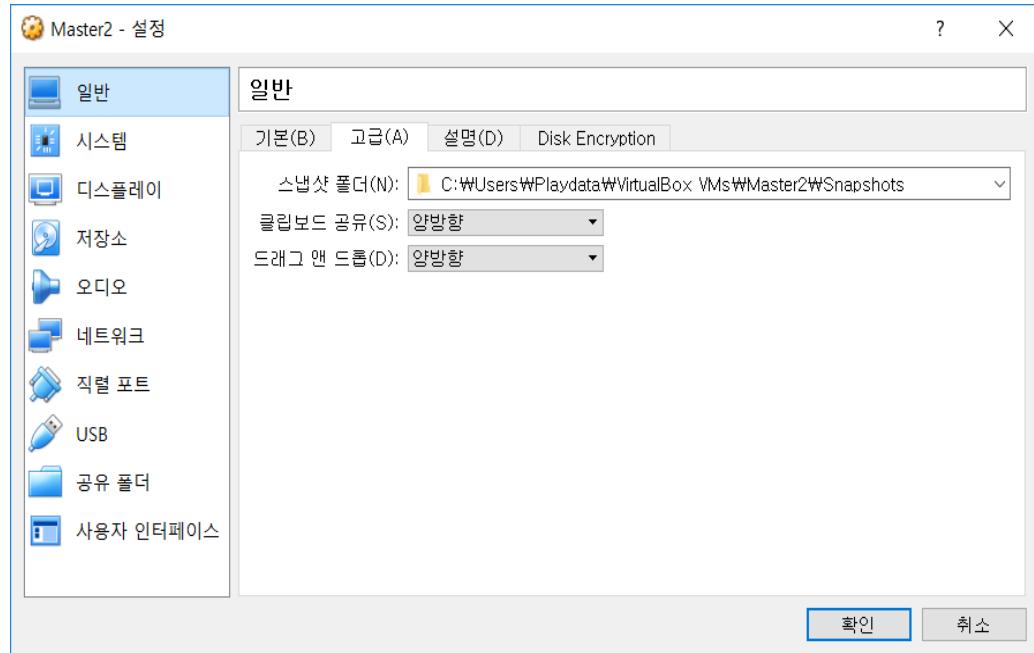


가상머신 생성

- 가상 머신 생성 후 설정 선택



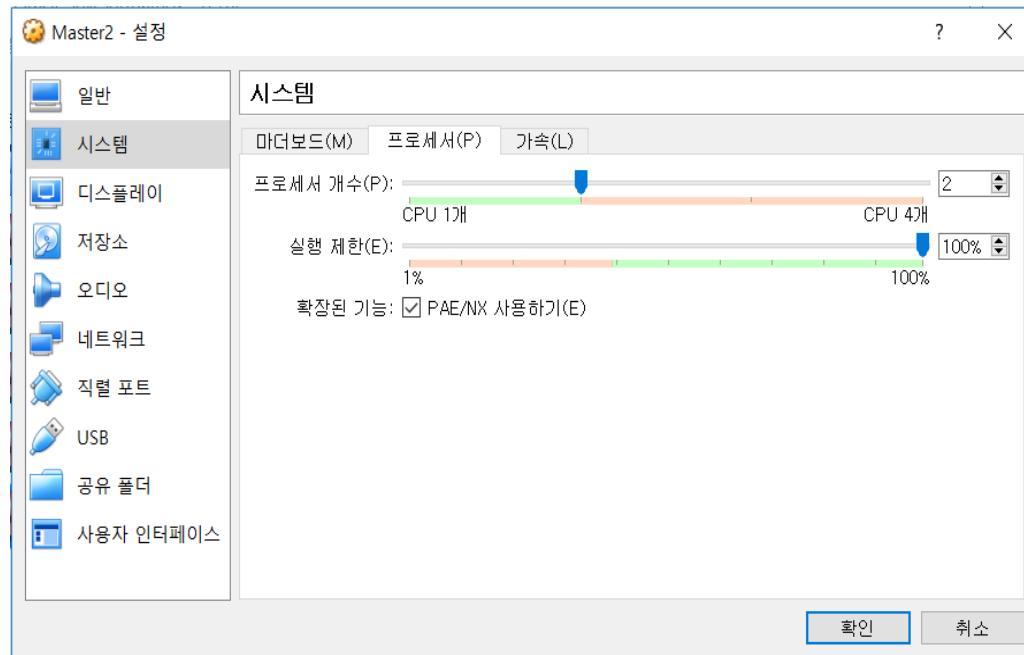
가상머신 생성



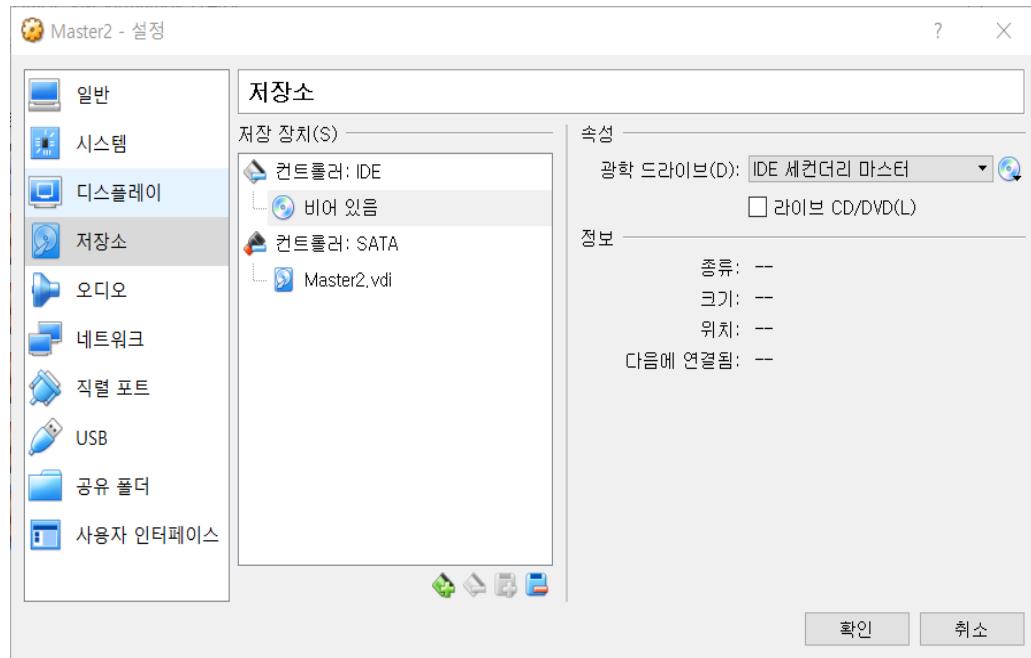
- 설정 선택 후
- 일반 메뉴 선택 -> 고급 탭 ->
- 클립보드 공유, 드래그 앤 드롭 모두 **양방향** 선택

가상머신 생성

- 시스템 -> 프로세서 탭 ->
- 프로세서 개수 : 녹색 부분에서 최대치 (2개)



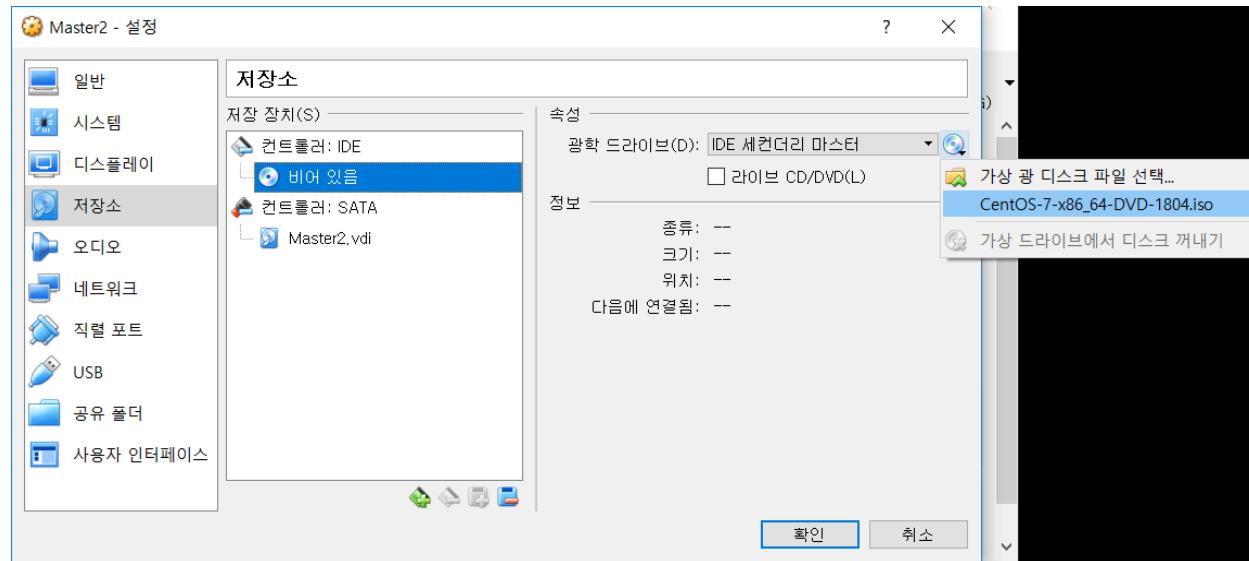
가상머신 생성



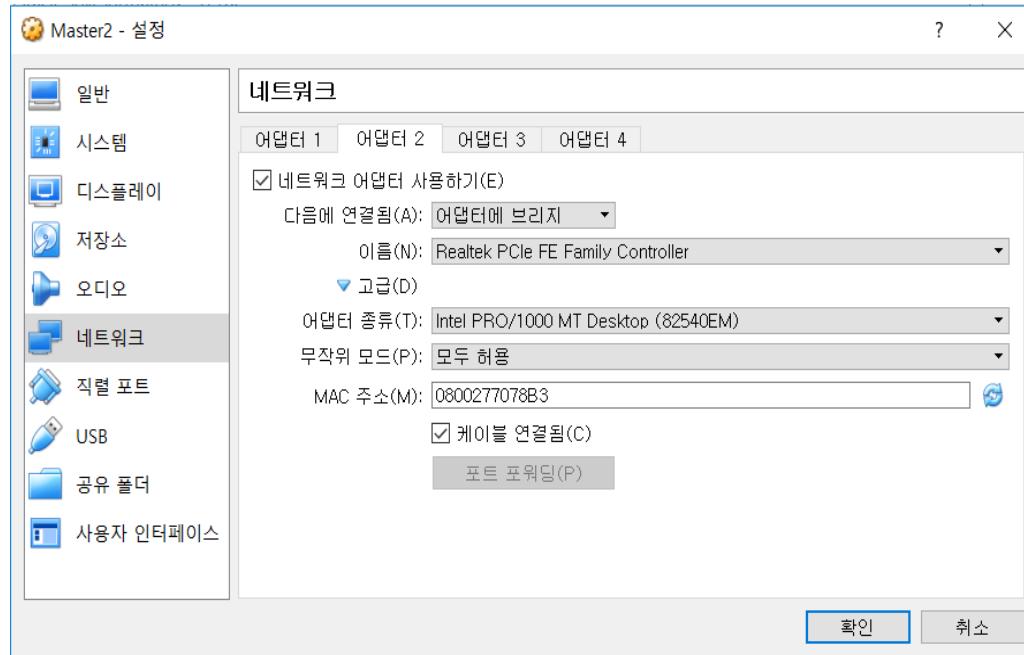
- 저장소 -> 컨트롤러: IDE 밑에 있는 **비어있음 선택** ->
- 우측에 있는 속성 밑에 있는 광학 드라이브(D): IDE 세컨더리 마스터 옆에 있는 **디스크 모양 아이콘 선택**

가상머신 생성

- 디스크 모양 아이콘 선택 후 보유하고 있는
- CentOS 7 iso 파일 선택

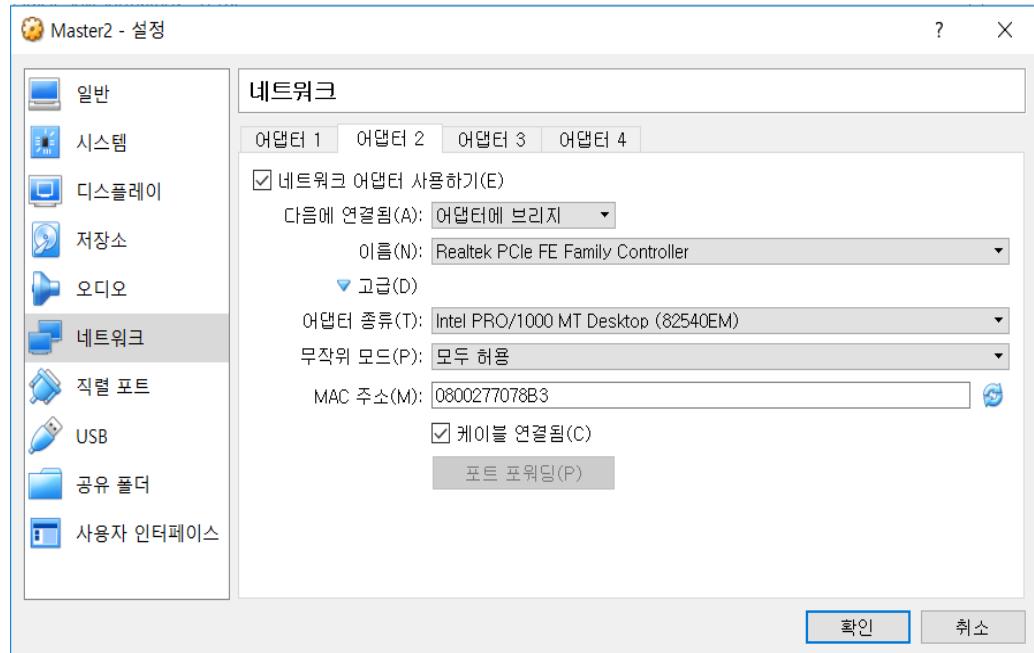


가상머신 생성



- 네트워크 -> 어댑터 1 ->
 - 다음에 연결됨 : NAT 선택
 - (어댑터 1 나머지 설정부분은 그대로)
-
- 현재 네트워크는 Windows OS에
 - 이더넷으로 연결되어 있는 상황 (LAN)

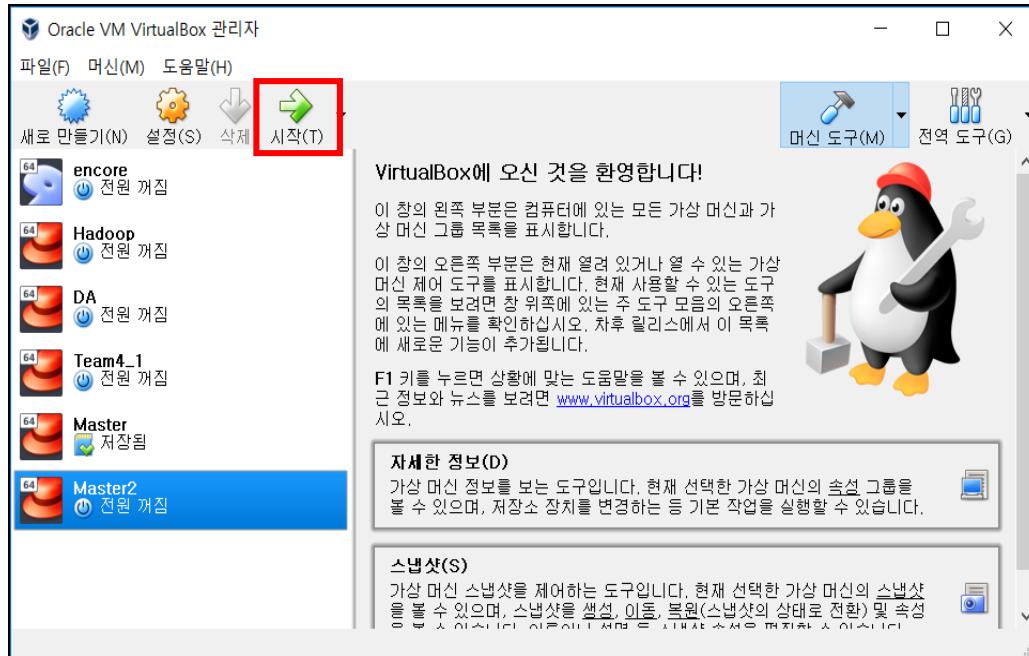
가상머신 생성



- 어댑터 2 ->
- 다음에 연결됨을 **어댑터에 브릿지**로 선택,
- 무작위 모드를 **모두 허용**으로 선택 후 확인

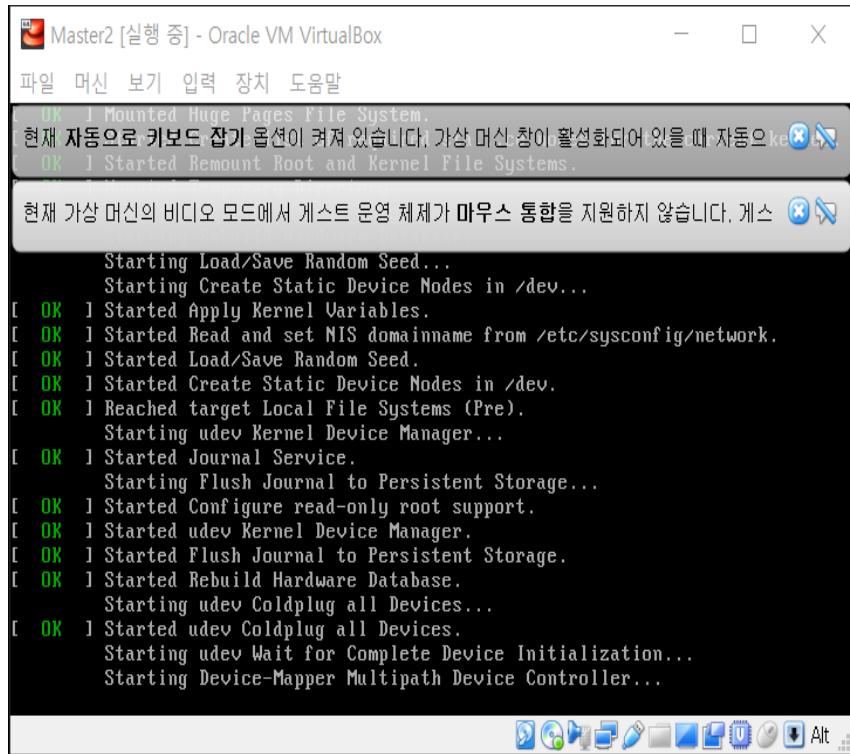
가상머신 생성

- 가상 머신 생성 후 **시작** 선택
- (나머지 설정 부분은 그대로 한 후 '만들기' 선택)



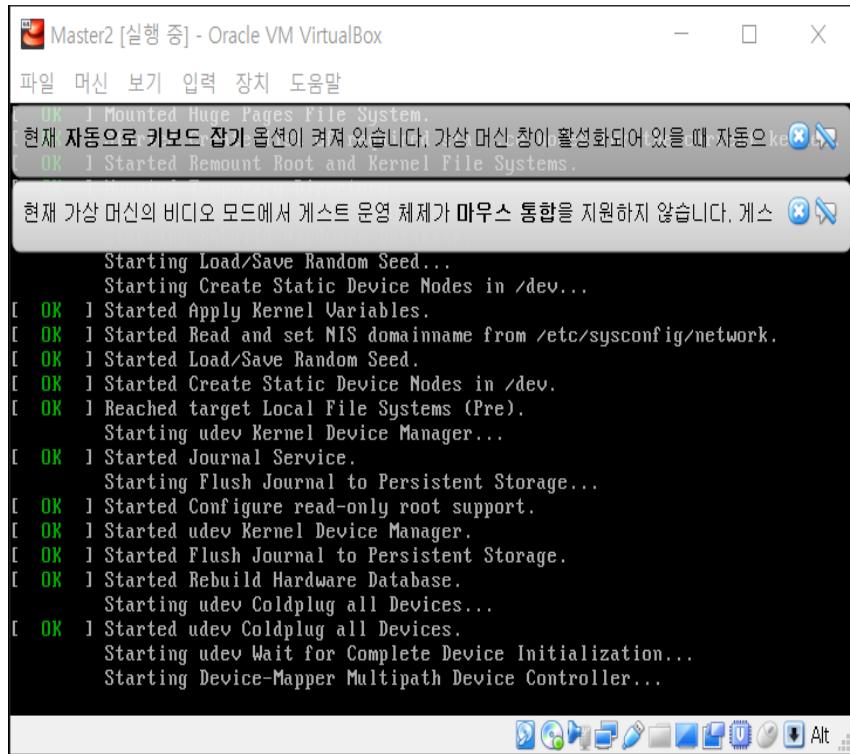
가상머신 생성

- 시작 후 Install 키보드를 이용해 Install CentOS 7 선택



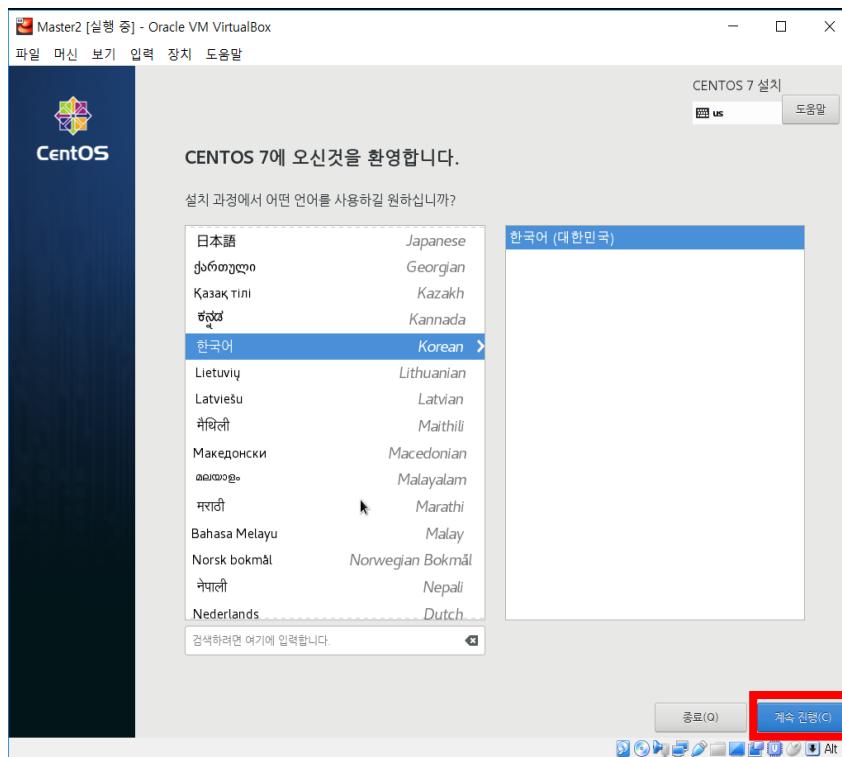
가상머신 생성

- 설치 진행 중



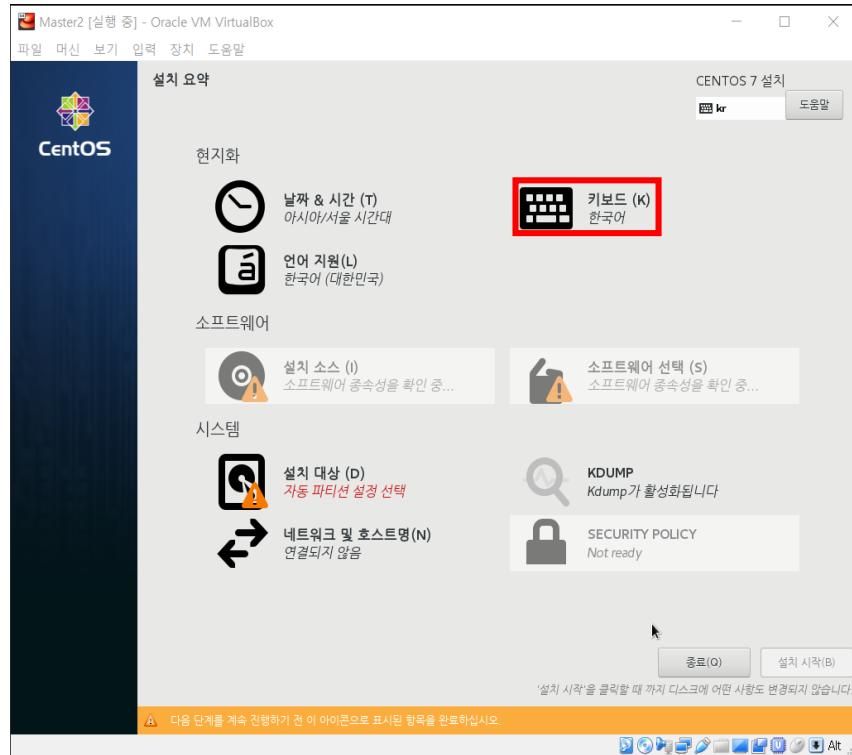
가상머신 생성

- 언어 설정 : 한국어 선택 후
- 계속 진행 선택



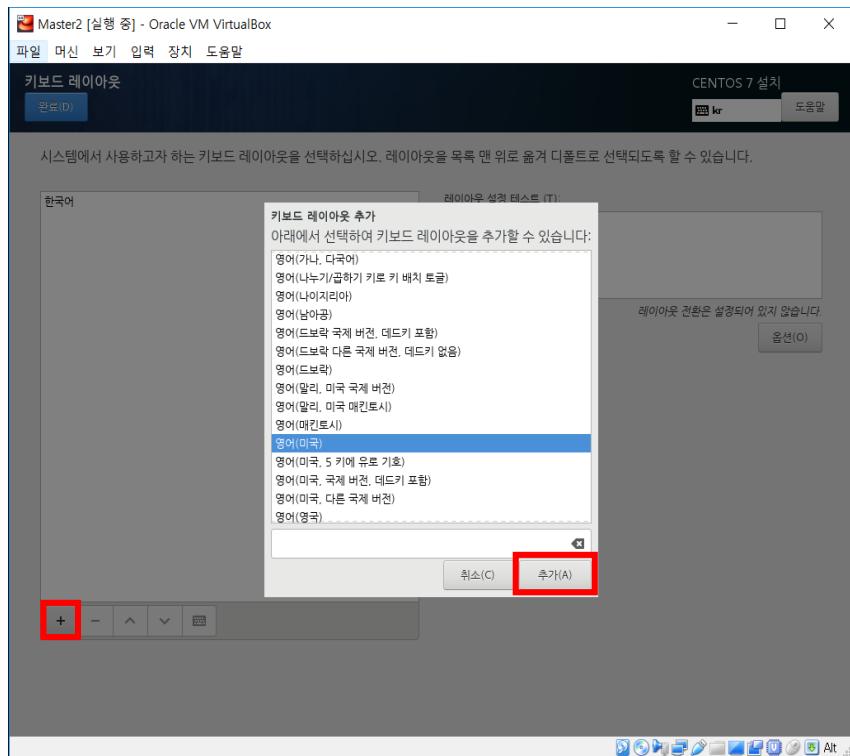
가상머신 생성

- 키보드 선택



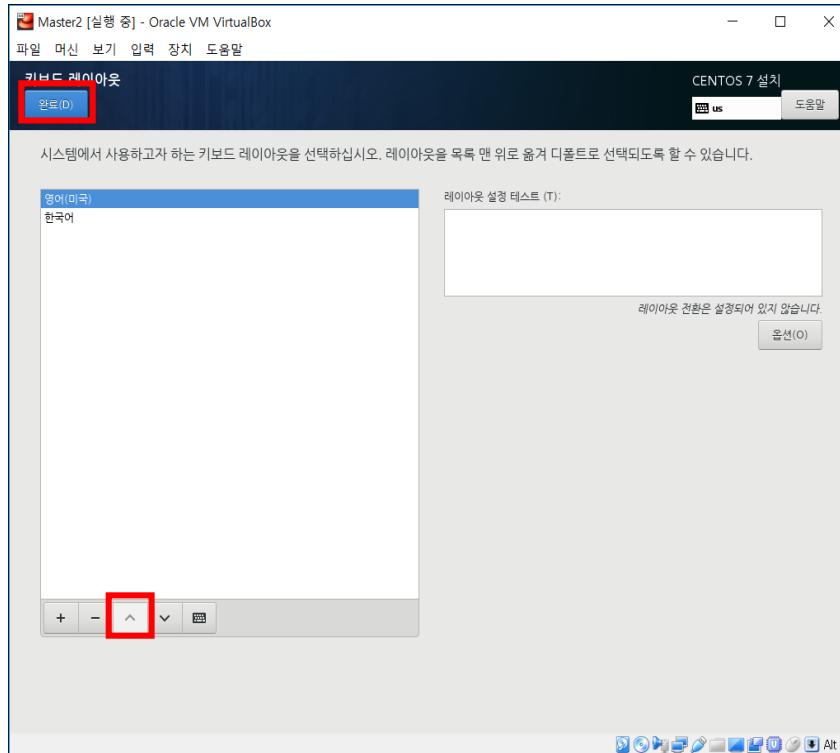
가상머신 생성

- "+" 선택
- 키보드 레이아웃 중 영어(미국) 선택 후 추가

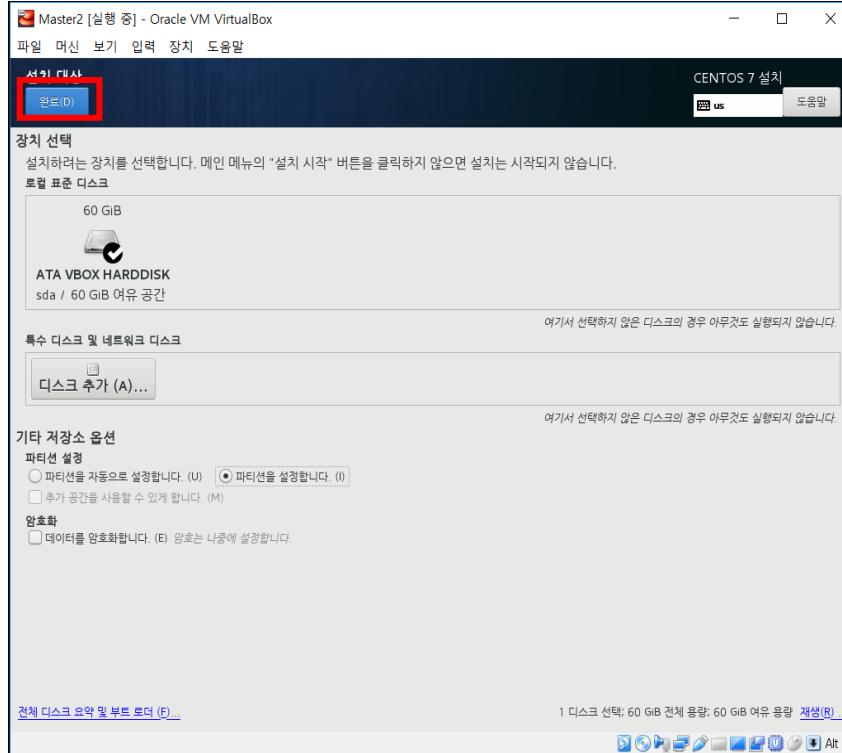


가상머신 생성

- 위로 이동 버튼을 이용해
- 영어(미국) 키보드 레이아웃을 한국어 위로 이동 후 완료 선택



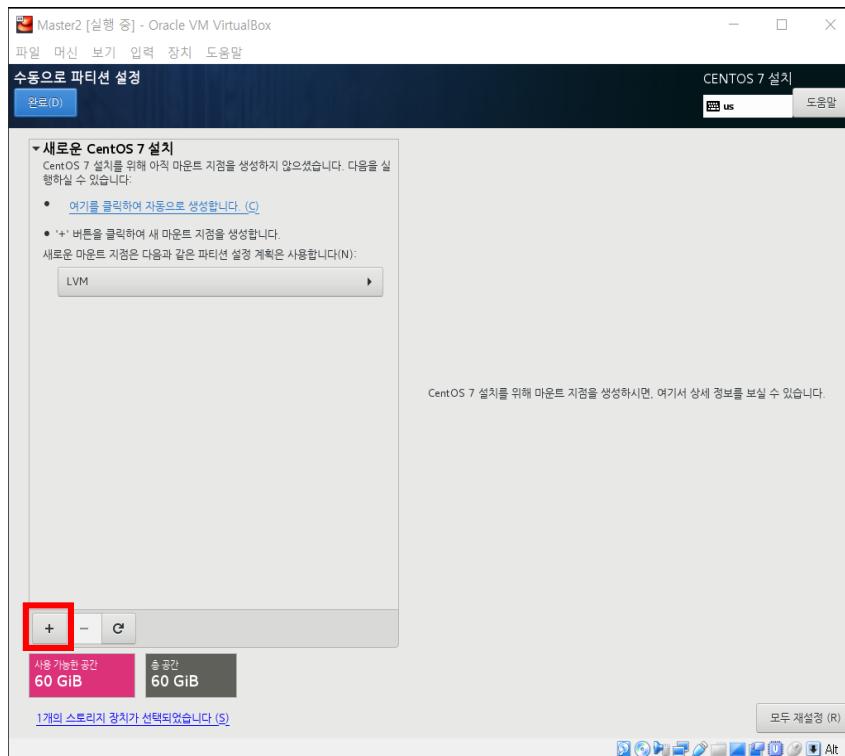
가상머신 생성



- ATA VBOX HARDDISK를 선택 ->
- 기타 저장소 옵션 밑에 있는
- **파티션을 설정합니다** 선택 후 완료

가상머신 생성

- 파티션을 직접 조정할 수 있는 화면
- "+" 버튼을 선택해 디렉토리 별로 용량을 직접 선택



가상머신 생성



- [/home] : 31.8 GiB
 - [/boot] : 200 MiB
 - [/] : 24 GiB
 - [swap] : 4096 MiB
- **입력 한 후 마운트 지점 추가 선택**
- [/home], [/boot], [/]는 장치 유형을 표준 파티션
 - 파일 시스템을 ext4로 선택 후 완료

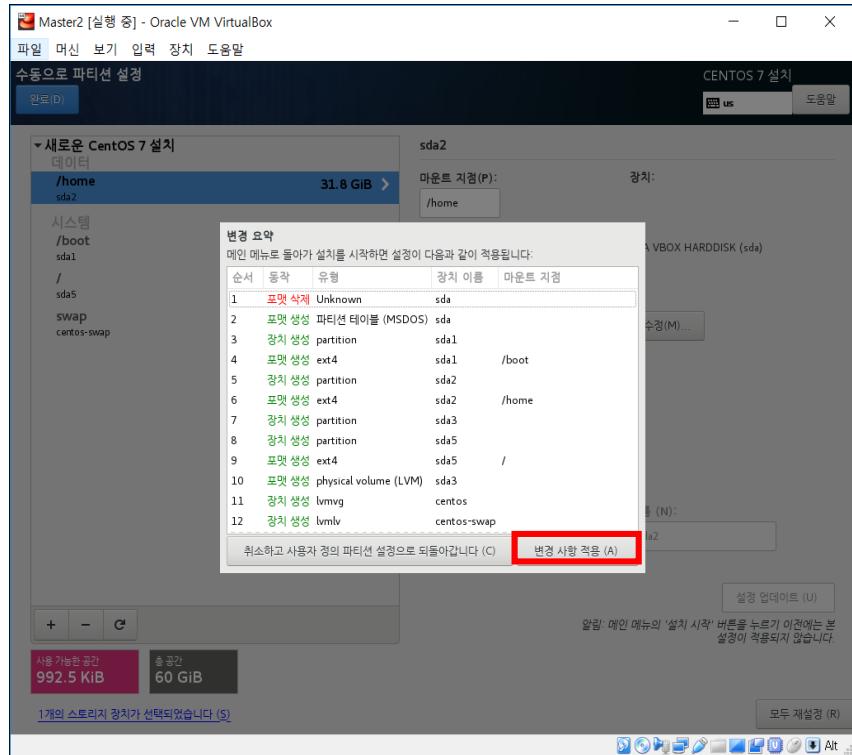
가상머신 생성

- [/home], [/boot], [/]는 장치 유형을 표준 파티션
- 파일 시스템을 ext4 로 선택 후 완료



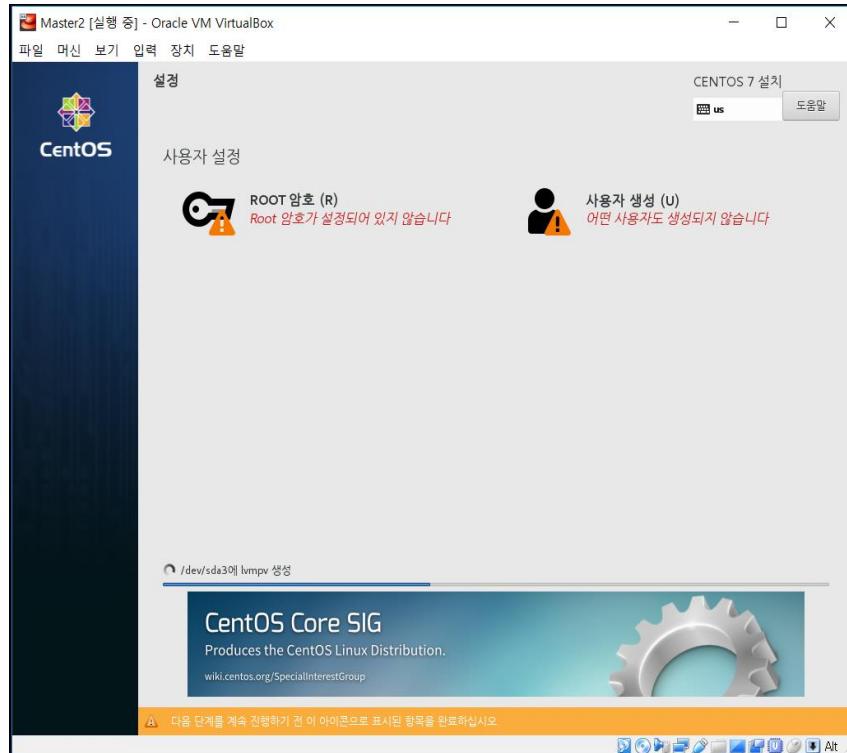
가상머신 생성

- 완료 후 변경 요약 창에서 **변경 사항 적용** 선택



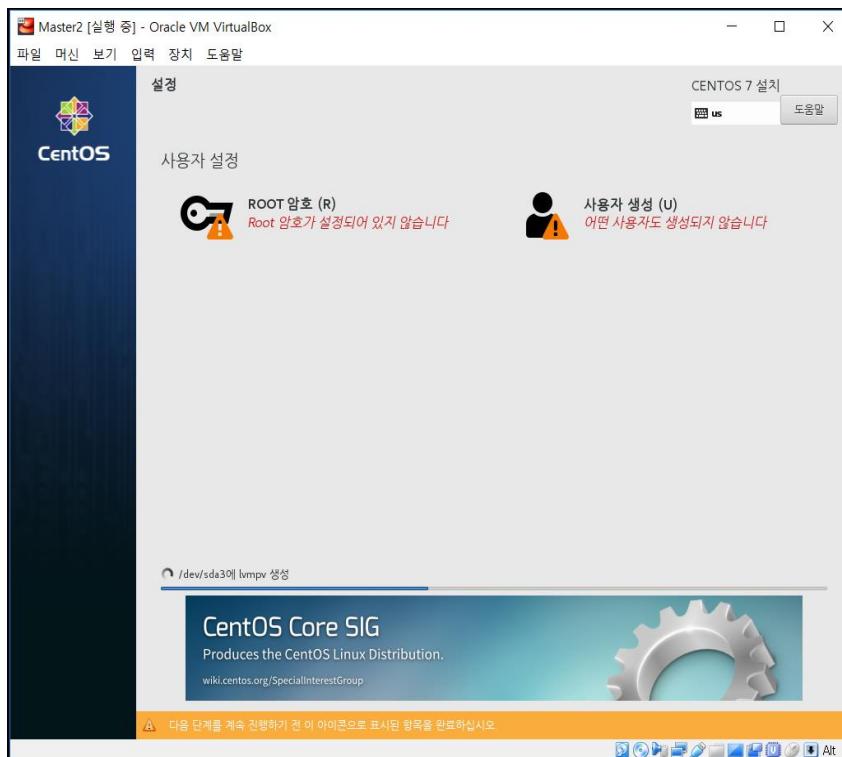
가상머신 생성

- 설치 시작 선택



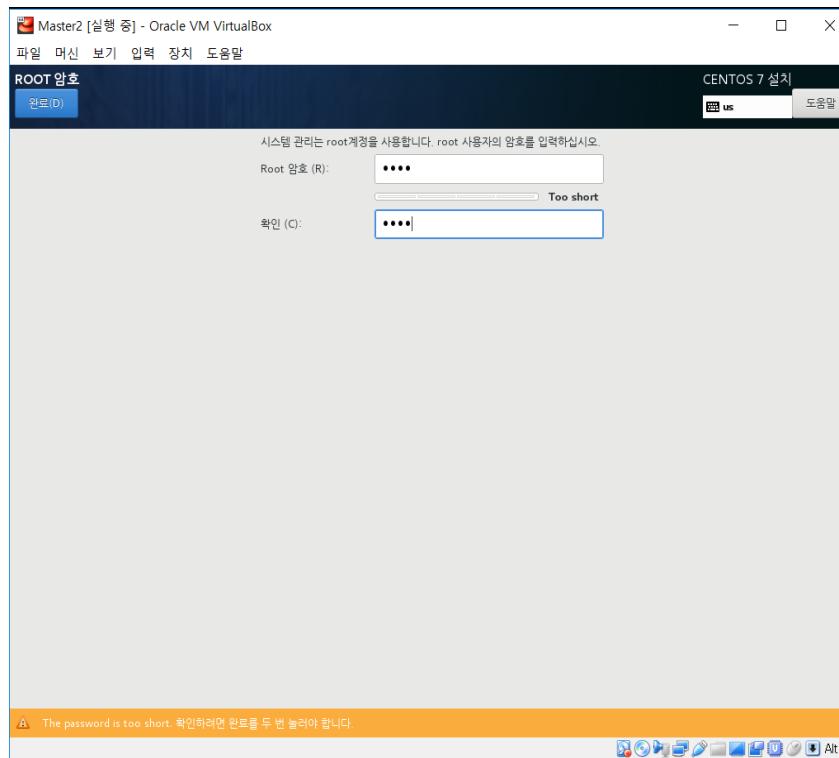
가상머신 생성

- Root 암호, 사용자 생성은
- 사용하는 본인이 알아서 진행하면 될 듯 합니다.



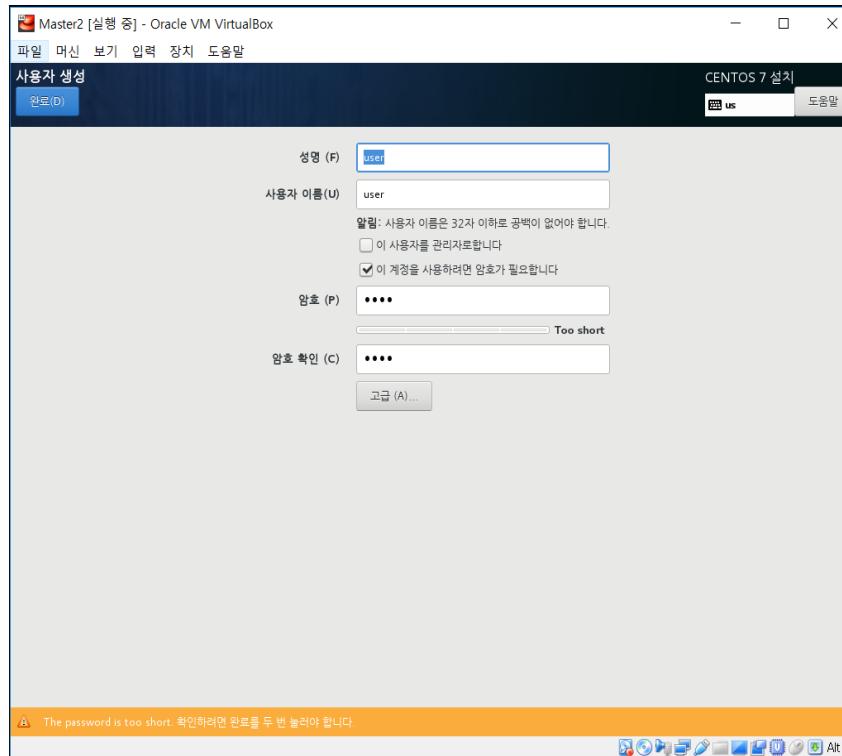
가상머신 생성

- Root 암호 메뉴 선택 후
- Root 계정에 대한 암호 설정



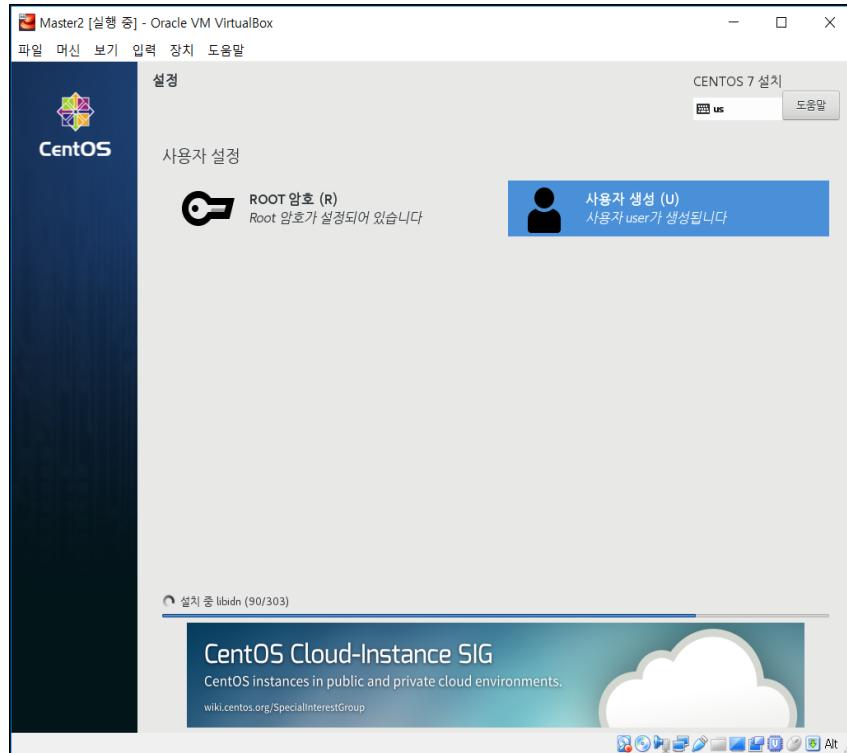
가상머신 생성

- 사용자 계정 추가 및 암호 설정 진행 후 완료 선택



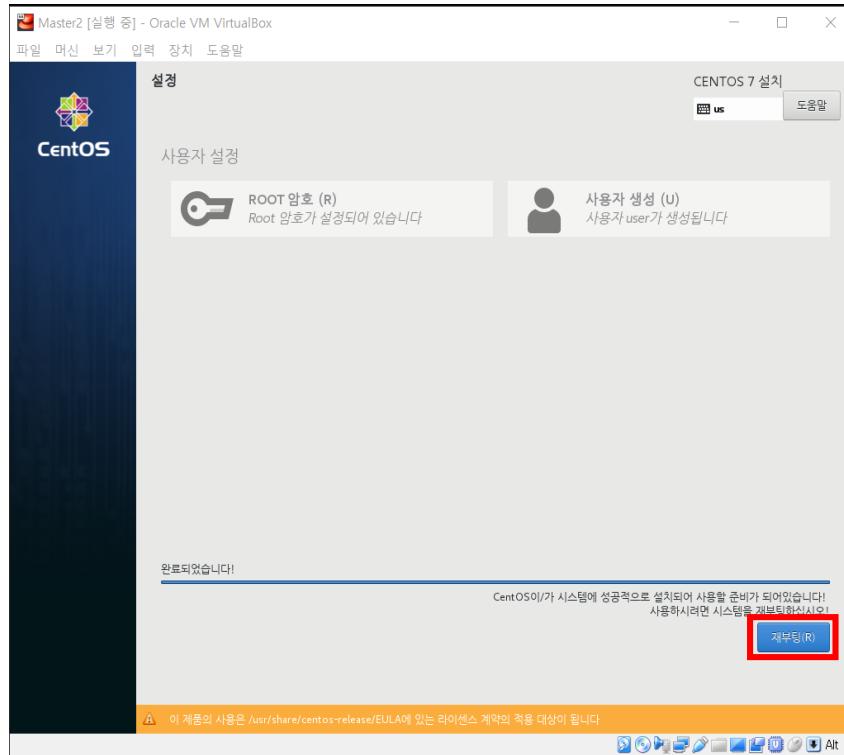
가상머신 생성

- 두 가지 설정 후 CentOS 7 설치가 계속 진행 됨



가상머신 생성

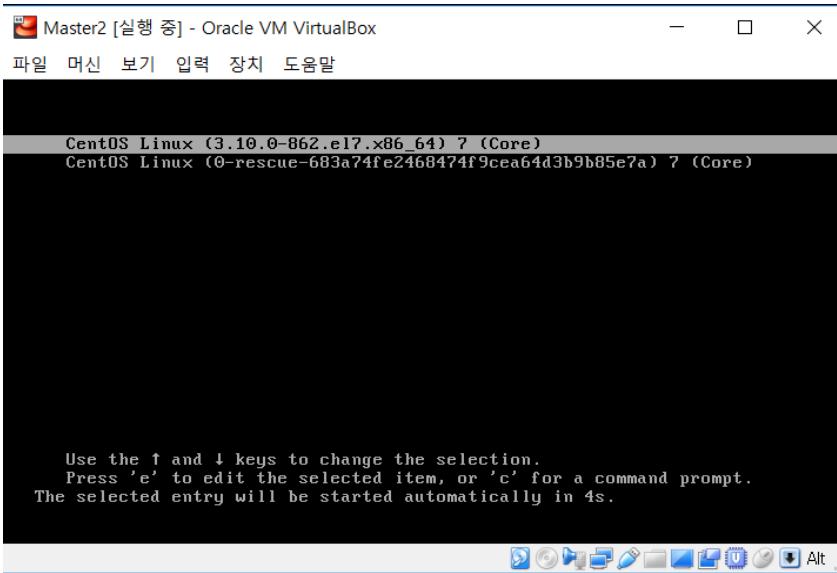
- 가상 머신이 생성이 완료된 후 **재부팅** 선택





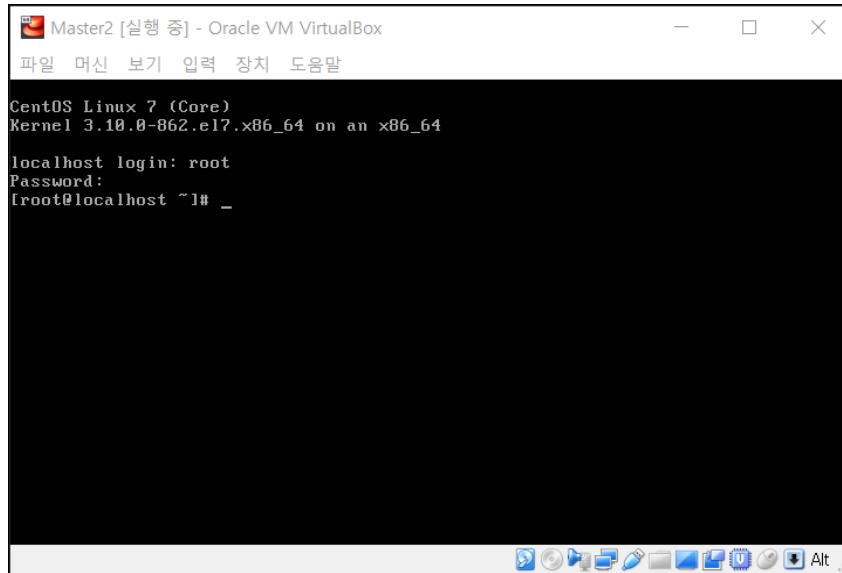
가상머신 실행

- 두 가지 항목 중
- CentOS Linux (3.10.0-862.e17.x86_64) 7 (Core) 선택



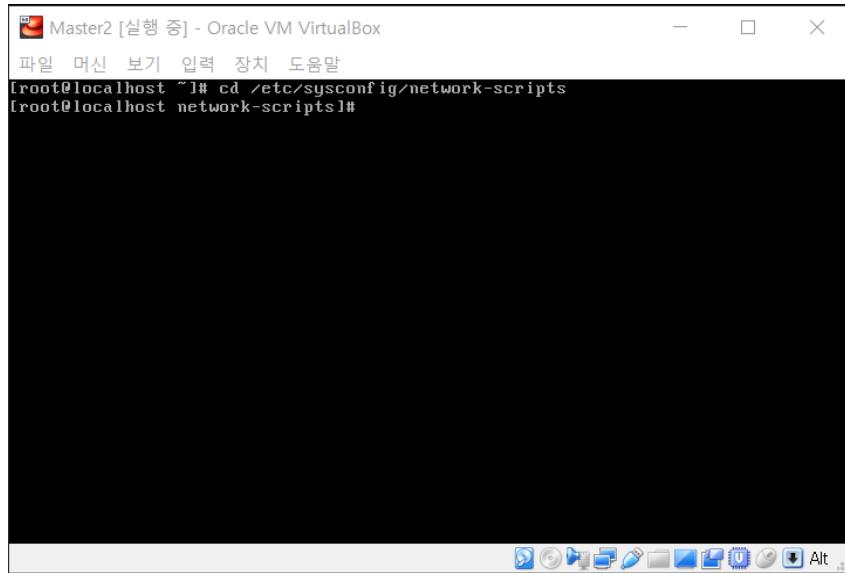
가상머신 실행

- Root 계정으로 CentOS 7에 로그인



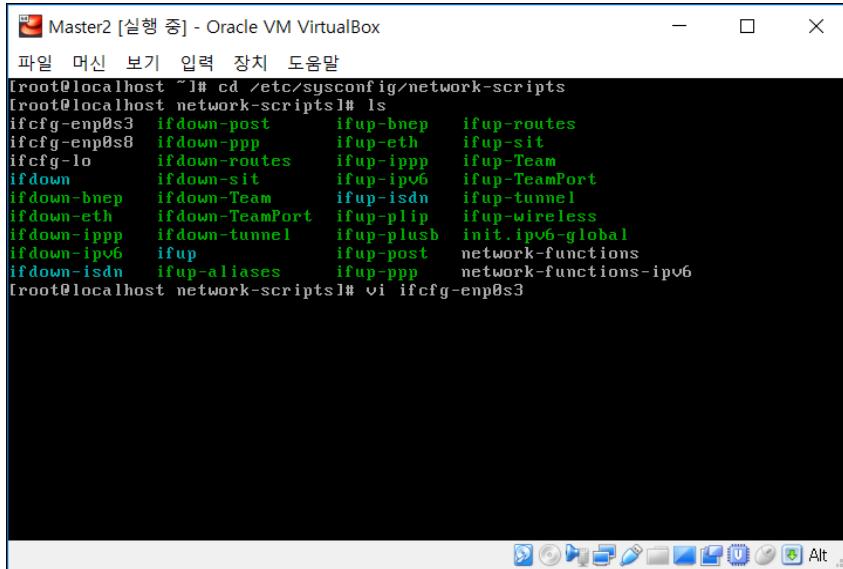
가상머신 실행

- 네트워크 설정을 위해 root ("~") 디렉토리에서
- /etc/sysconfig/network-scripts 디렉토리로 이동



가상머신 실행

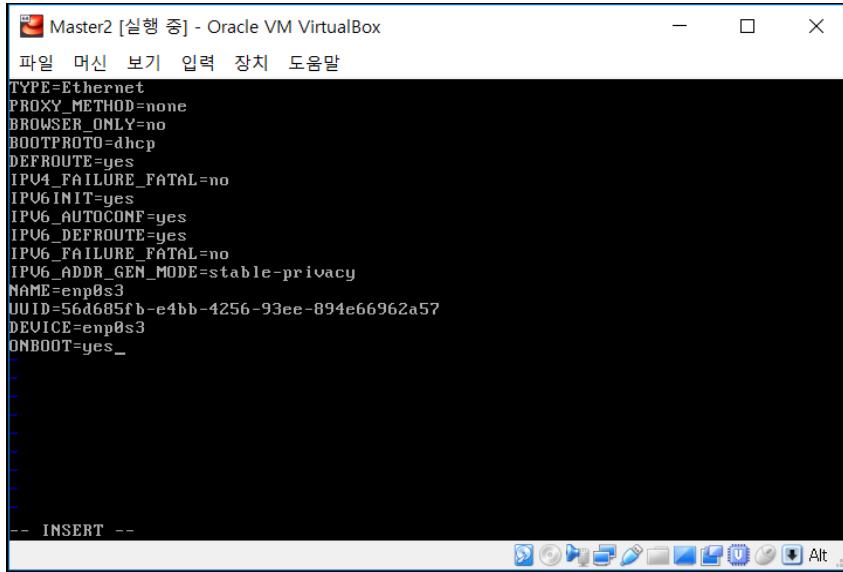
- vi ifcfg-enp0s3 실행



```
Master2 [실행 중] - Oracle VM VirtualBox
파일 머신 보기 입력 장치 도움말
[root@localhost ~]# cd /etc/sysconfig/network-scripts
[root@localhost network-scripts]# ls
ifcfg-enp0s3  ifdown-post    ifup-bnep   ifup-routes
ifcfg-enp0s8  ifdown-ppp     ifup-eth    ifup-sit
ifcfg-lo      ifdown-routes  ifup-ipp   ifup-Team
ifdown        ifdown-sit    ifup-ipv6  ifup-TeamPort
ifdown-bnep   ifdown-Team   ifup-isdn  ifup-tunnel
ifdown-eth    ifdown-TeamPort ifup-plip  ifup-wireless
ifdown-ipp   ifdown-tunnel  ifup-plusb init.ipv6-global
ifdown-ipv6   ifup         ifup-post  network-functions
ifdown-isdn   ifup-aliases  ifup-ppp   network-functions-ipv6
[root@localhost network-scripts]# vi ifcfg-enp0s3
```

가상머신 실행

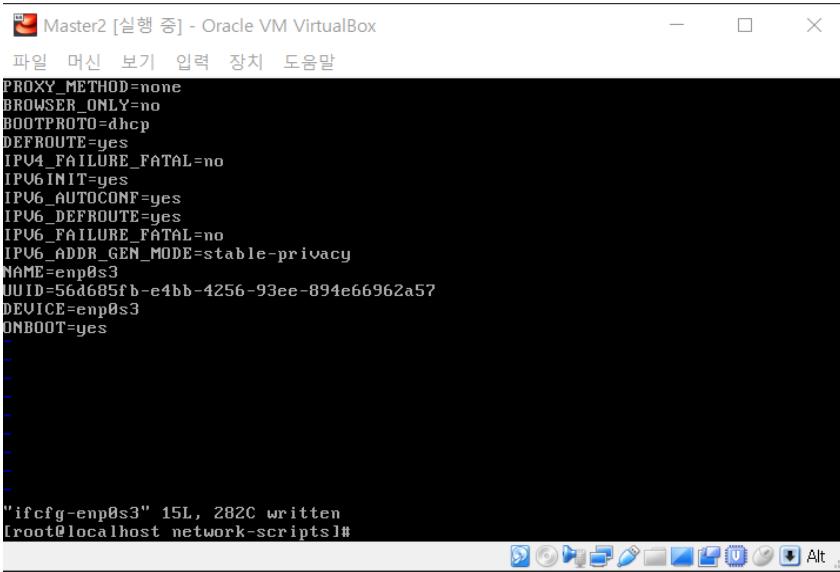
- ONBOOT의 항목을 no에서 yes로 변경한 후 저장
- 수정하기 위해 insert 실행
- insert는 키보드로 "i"를 실행하면 가능



- 수정 및 작성한 후
- 저장하는 명령 **esc -> : (콜론) -> wq**

가상머신 실행

- ONBOOT의 항목을 no에서 yes로 변경한 후 저장
- 수정하기 위해 insert 실행
- insert는 키보드로 "i"를 실행하면 가능



The screenshot shows a terminal window with the title "Master2 [실행 중] - Oracle VM VirtualBox". The window contains the following text:

```
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp0s3
UUID=56d685fb-e4bb-4256-93ee-894e66962a57
DEVICE=enp0s3
ONBOOT=yes

ifcfg-enp0s3" 15L, 282C written
[root@localhost network-scripts]#
```

The terminal window has a dark background and light-colored text. The bottom of the window shows a standard Linux desktop environment toolbar with icons for file operations like cut, copy, paste, and a terminal icon.

- 수정 및 작성한 후
- 저장하는 명령 **esc -> : (콜론) -> wq**

가상머신 실행

- network 서비스를 재시작하기 위해
- [root@localhost network-scripts]# systemctl restart network.service 입력

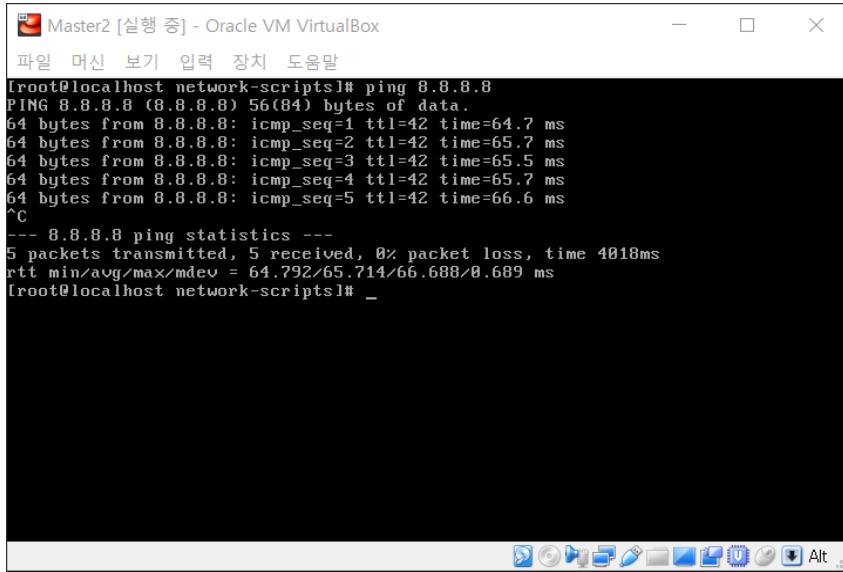
The screenshot shows a terminal window titled "Master2 [실행 중] - Oracle VM VirtualBox". The window contains the following text:

```
파일 머신 보기 입력 장치 도움말
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp0s3
UUID=56d685fb-e4bb-4256-93ee-894e66962a57
DEVICE=enp0s3
ONBOOT=yes

"ifcfg-enp0s3" 15L, 282C written
[root@localhost network-scripts]# systemctl restart network.service
```

가상머신 실행

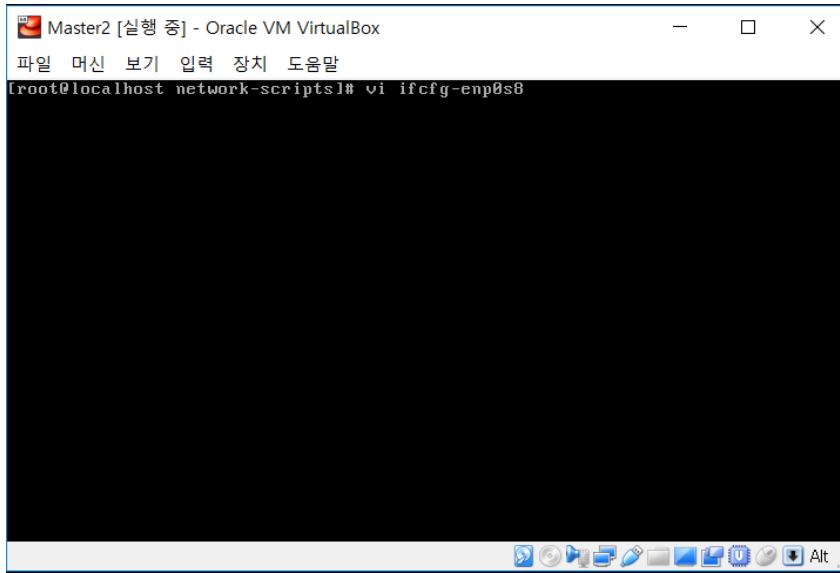
- network 가동 확인을 위해 ping 실행
- [root@localhost network-scripts]# ping 8.8.8.8 입력



```
[root@localhost network-scripts]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=42 time=64.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=42 time=65.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=42 time=65.5 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=42 time=65.7 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=42 time=66.6 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4018ms
rtt min/avg/max/mdev = 64.792/65.714/66.688/0.689 ms
[root@localhost network-scripts]# _
```

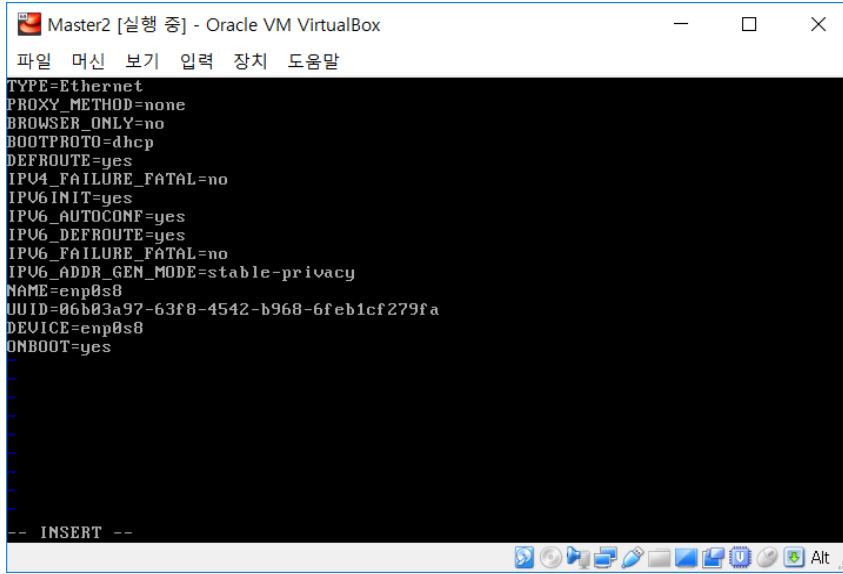
가상머신 실행

- 고정 IP 설정
- [root@localhost network-scripts]# vi ifcfg-enp0s8



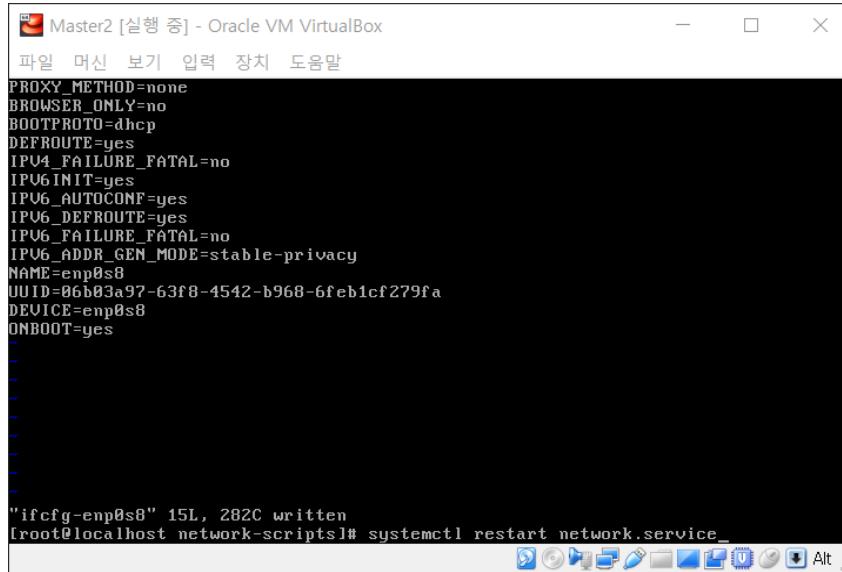
가상머신 실행

- [root@localhost network-scripts]# vi ifcfg-enp0s8
- ONBOOT부분에서 no를 yes로 변경 후 저장



가상머신 실행

- 네트워크 서비스 재시작
- [root@localhost network-scripts]# systemctl restart network.service



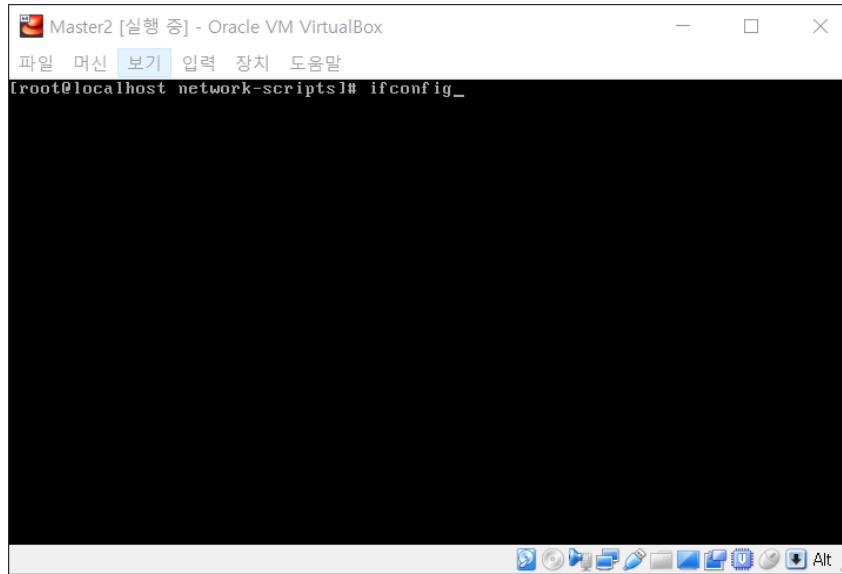
The screenshot shows a terminal window titled "Master2 [실행 중] - Oracle VM VirtualBox". The window contains the following text:

```
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp0s8
UUID=06b03a97-63f8-4542-b968-6feb1cf279fa
DEVICE=enp0s8
ONBOOT=yes

"ifcfg-enp0s8" 15L, 282C written
[root@localhost network-scripts]# systemctl restart network.service
```

가상머신 실행

- ip 확인하기 위해
- [root@localhost network-scripts]# ifconfig 입력



가상머신 실행

- ip 확인하기 위해
- [root@localhost network-scripts]# ip addr 입력

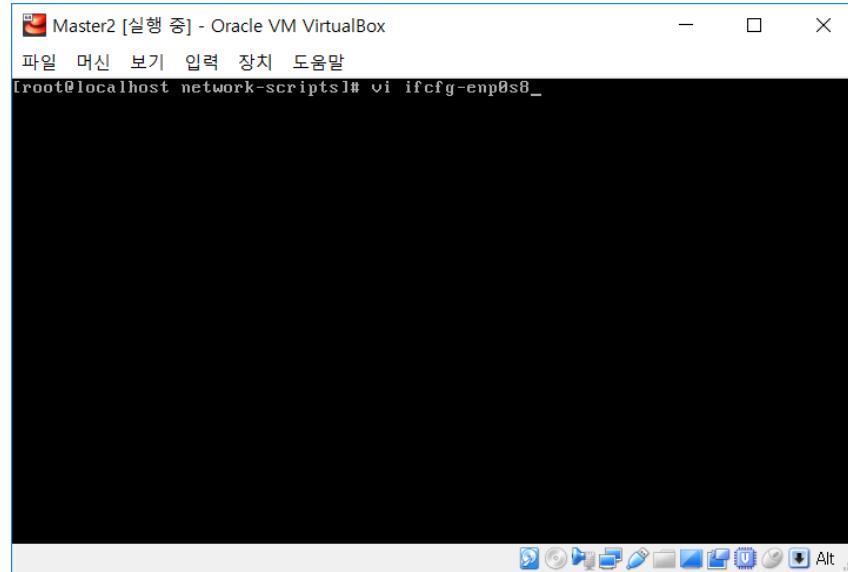
```
Master2 [실행 중] - Oracle VM VirtualBox
파일 머신 보기 입력 장치 도움말
ether 00:00:27:3b:94:e1 txqueuelen 1000 (Ethernet)
RX packets 38 bytes 7543 (7.3 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 92 bytes 8124 (7.9 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.103.53 netmask 255.255.255.0 broadcast 192.168.103.255
inet6 fe80::ac11:1b1b:d92:f729 prefixlen 64 scopeid 0x20<link>
ether 00:00:27:70:b3 txqueuelen 1000 (Ethernet)
RX packets 197 bytes 21266 (20.7 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 116 bytes 16077 (15.7 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 8 bytes 668 (668.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 8 bytes 668 (668.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

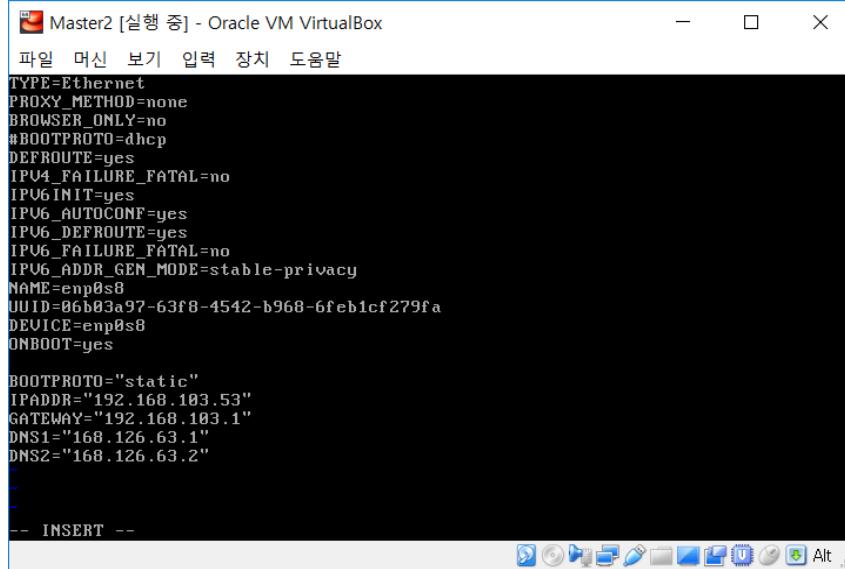
[root@localhost network-scripts]# _
```

가상머신 실행



- 현재 ifcfg-enp0s8에 할당 받은
- ip는 192.168.103.53
- 이 ip를 고정 ip로 설정하기 위해 다시
- [root@localhost network-scripts]# vi ifcfg-enp0s8 입력

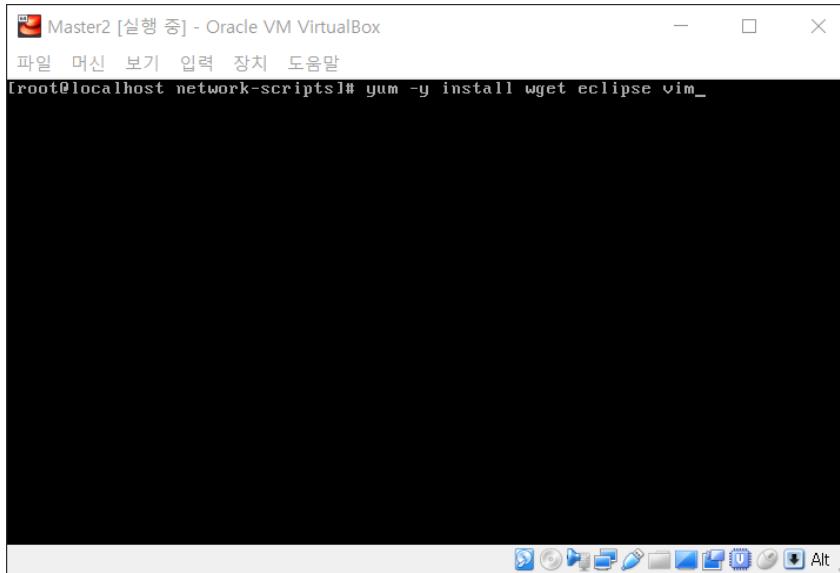
가상머신 실행



- 외부에서 접속하기 위해 고정 IP 설정
- 기존 BOOTPROTO=dhcp를 주석처리하고
- ONBOOT 하단에 다음과 같이 작성
- BOOTPROTO="static"
- IPADDR="잠시 전 ifcfg-enp0s8에 할당받은 ip 입력"
- GATEWAY="ifcfg-enp0s8에 할당받은 ip 자리 중 앞 세 위치 자리 값 입력" - ex) 192.168.103.1
- DNS1="168.126.63.1"
- DNS2="168.126.63.2"
- 입력 후 저장

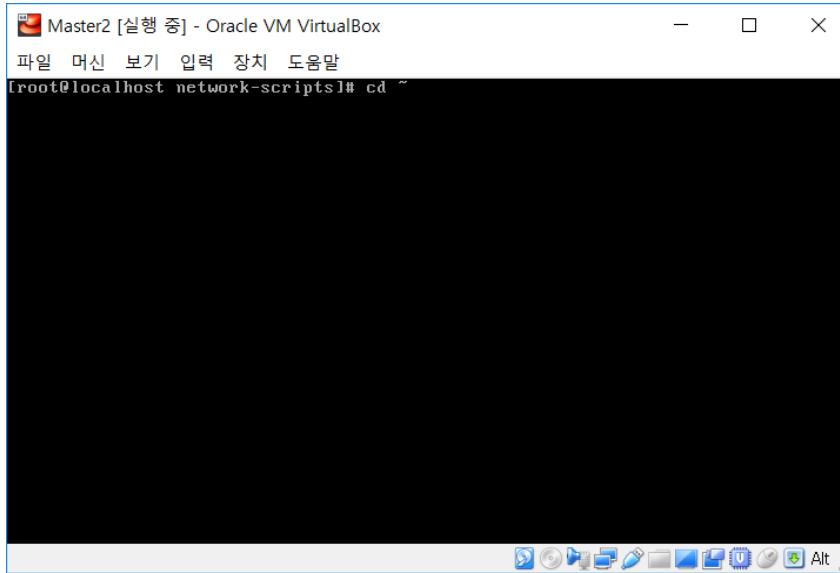
가상머신 실행

- 필요한 프로그램 설치
- [root@localhost network-scripts]# yum -y install wget vim

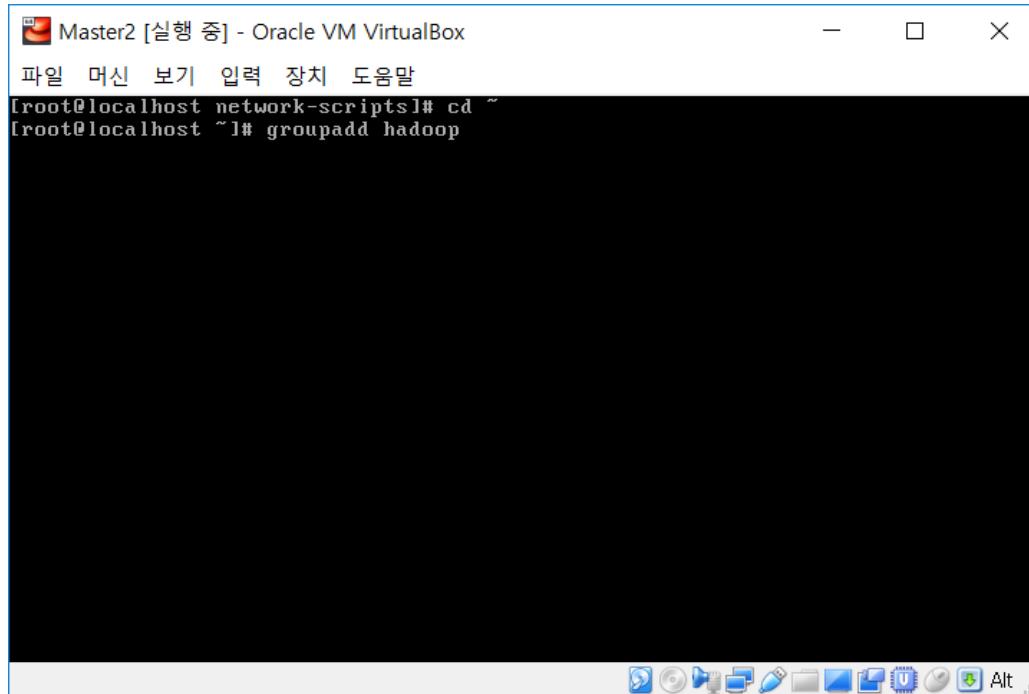


가상머신 실행

- ~ 디렉토리로 이동
- [root@localhost network-scripts]# cd ~



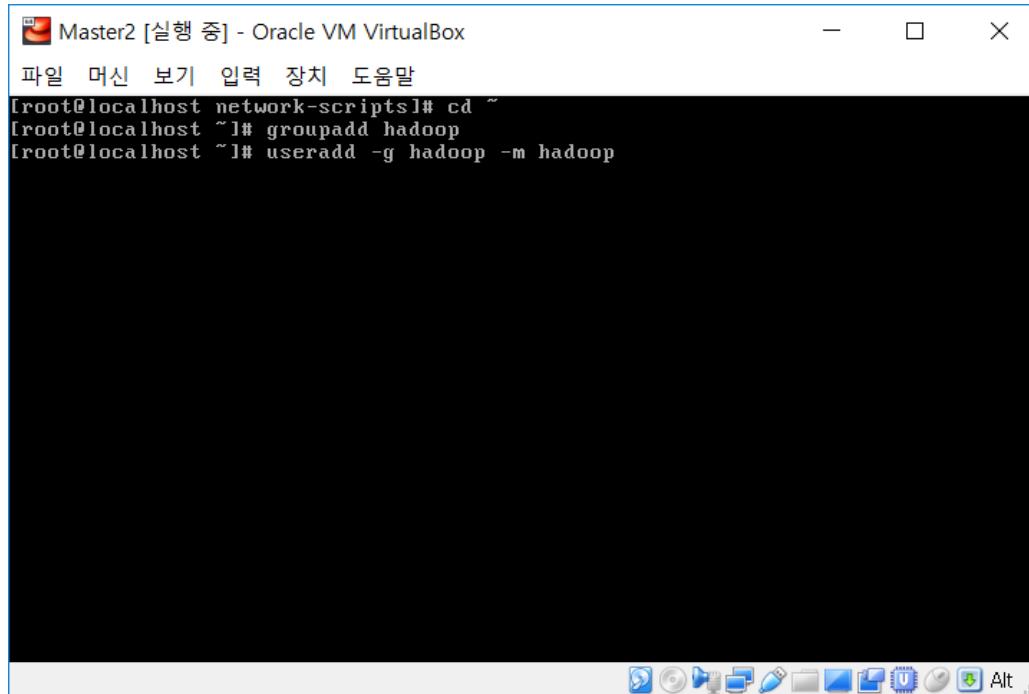
가상머신 실행



hadoop 그룹 생성

```
[root@localhost ~]# groupadd hadoop
```

가상머신 실행



Master2 [실행 중] - Oracle VM VirtualBox

파일 머신 보기 입력 장치 도움말

```
[root@localhost network-scripts]# cd ~  
[root@localhost ~]# groupadd hadoop  
[root@localhost ~]# useradd -g hadoop -m hadoop
```

hadoop 그룹에 속하는 hadoop 이름의 사용자 계정 생성

```
[root@localhost ~]# useradd -g hadoop -m hadoop
```

가상머신 실행

```
[root@localhost network-scripts]# cd ~  
[root@localhost ~]# groupadd hadoop  
[root@localhost ~]# useradd -g hadoop -m hadoop  
[root@localhost ~]# passwd hadoop  
Changing password for user hadoop.  
New password:  
BAD PASSWORD: The password is shorter than 8 characters  
Retype new password:  
passwd: all authentication tokens updated successfully.  
[root@localhost ~]#
```

hadoop 계정의 password 생성

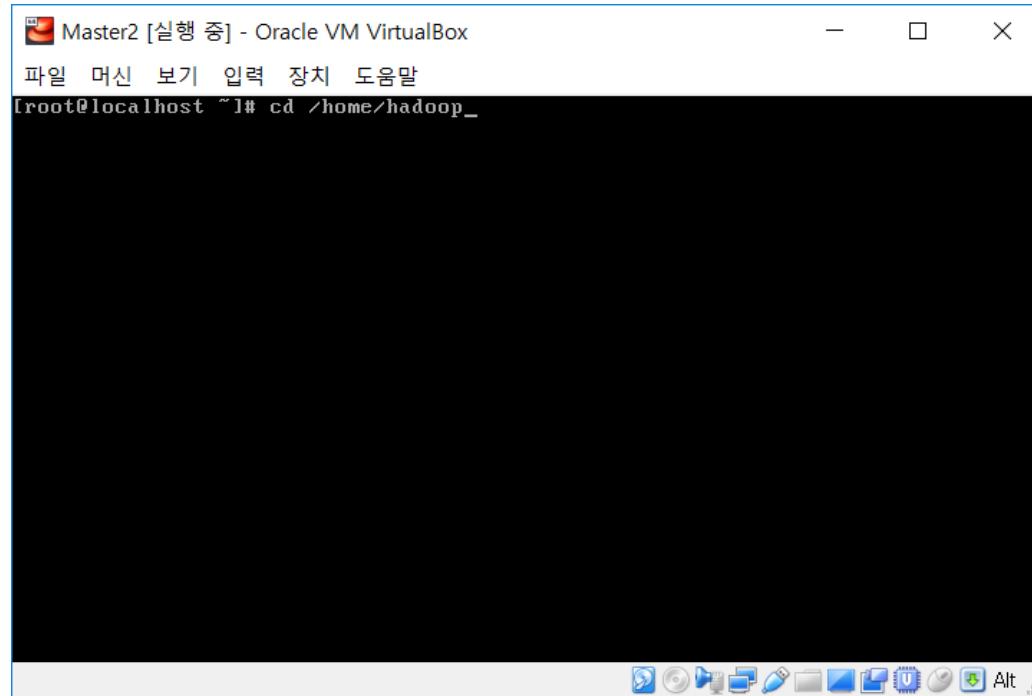
X

[root@localhost ~]# passwd hadoop

이후 본인이 hadoop 계정에 생성할 password 입력 (확인 입력까지 총 2번)

hadoop 계정의 password
생성 하지 말기

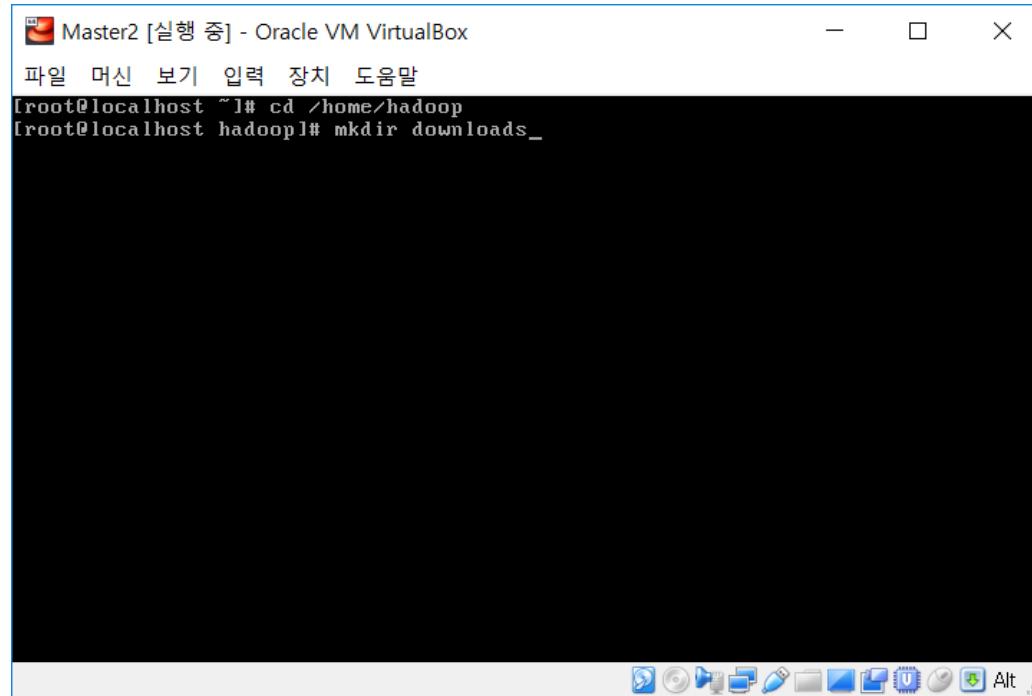
가상머신 실행



hadoop 디렉토리 이동

```
[root@localhost ~]# cd /home/hadoop
```

가상머신 실행

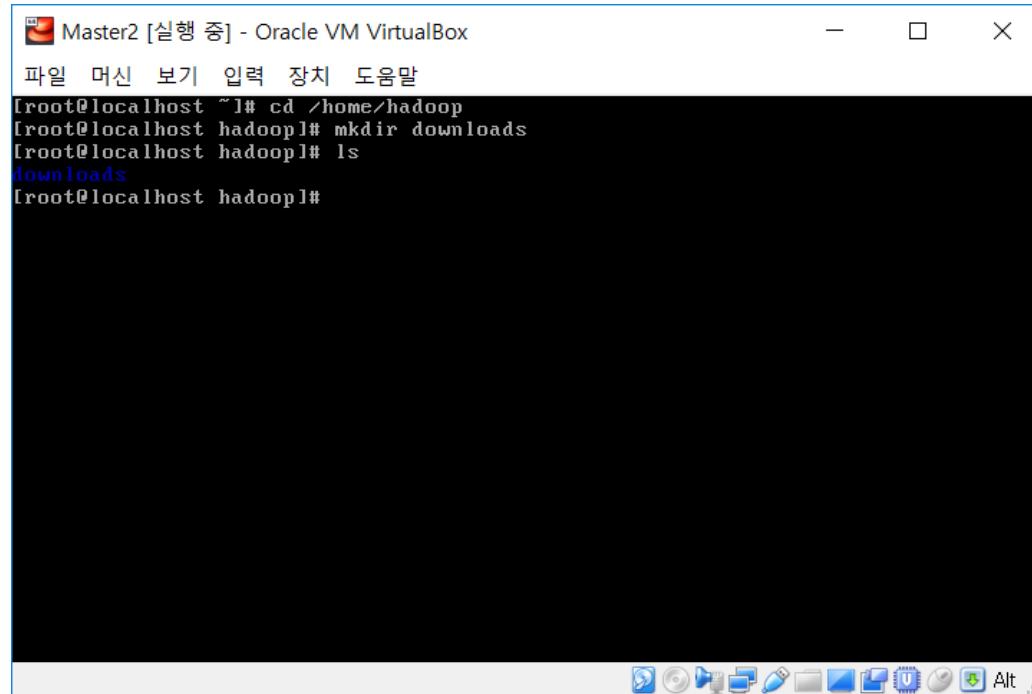


```
Master2 [실행 중] - Oracle VM VirtualBox
파일 머신 보기 입력 장치 도움말
[root@localhost ~]# cd /home/hadoop
[root@localhost hadoop]# mkdir downloads_
```

hadoop 디렉토리 하위에 downloads 디렉토리 생성

```
[root@localhost hadoop]# mkdir downloads
```

가상머신 실행

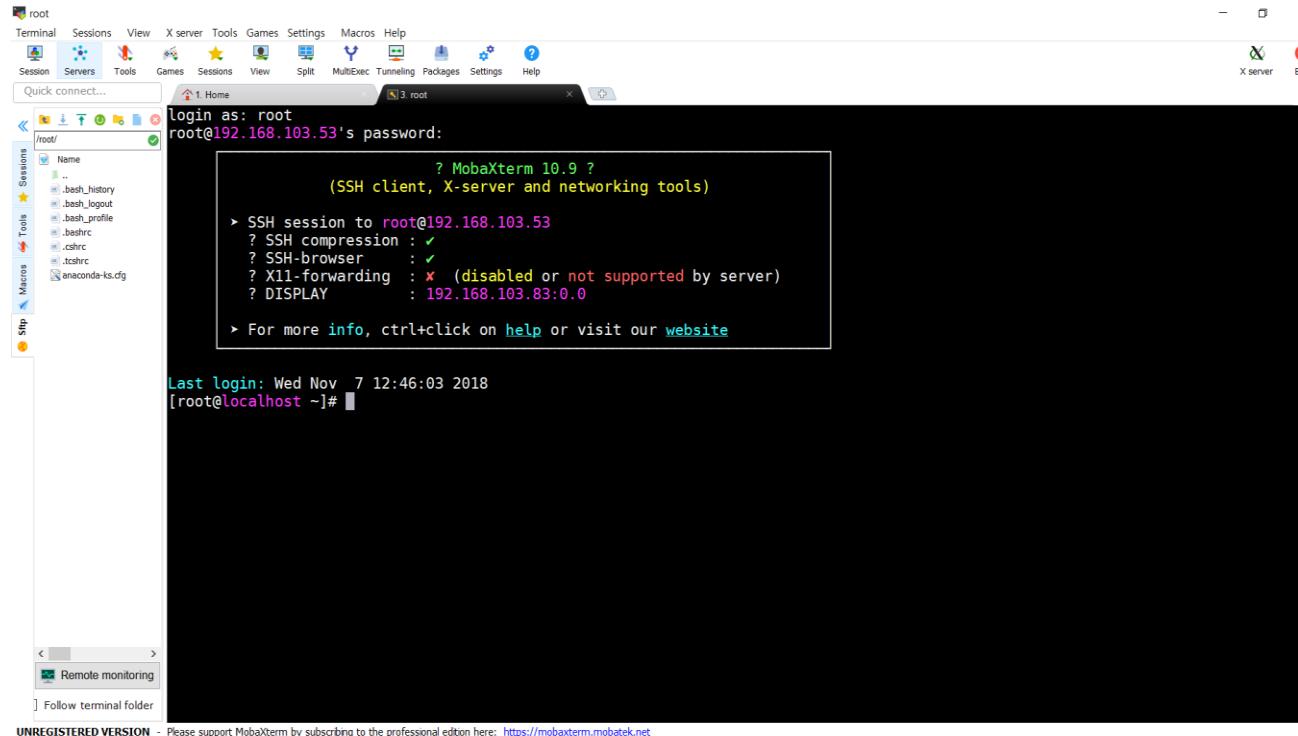


```
Master2 [실행 중] - Oracle VM VirtualBox
파일 머신 보기 입력 장치 도움말
[root@localhost ~]# cd /home/hadoop
[root@localhost hadoop]# mkdir downloads
[root@localhost hadoop]# ls
downloads
[root@localhost hadoop]#
```

hadoop 디렉토리 하위에 downloads 디렉토리 생성

```
[root@localhost hadoop]# mkdir downloads
```

가상머신 실행



MobaXterm으로 가상 머신 접속

가상머신 실행

The screenshot shows the Oracle Java SE Archive Downloads page. At the top, there's a navigation bar with links for Overview, Downloads, Documentation, Community, Technologies, and Training. On the left, a sidebar lists categories like Java SE, Java EE, Java ME, Java Subscription, Java Embedded, Java Card, Java TV, Community, and Java Magazine. The main content area is titled "Java SE 7 Archive Downloads" and contains a message about downloading the Java Platform, Standard Edition Development Kit (JDK). It includes sections for "Java Resources" and "Java SDKs and Tools". A table at the bottom lists various Java SE Development Kit 7u67 download options, including Linux x86, Linux x86_64, Solaris SPARC, Solaris SPARC 64-bit, Solaris x64, Solaris x86, Solaris x86_64, Windows x86, and Windows x64. Each row has a "Download" link.

Jdk 1.7 (jdk-7u67-linux-x64.tar.gz)을 받기 위해
오라클 홈페이지에서 다운로드 받은 후
"Filezilla"를 통해 /home/hadoop/downloads에 업로드

Url

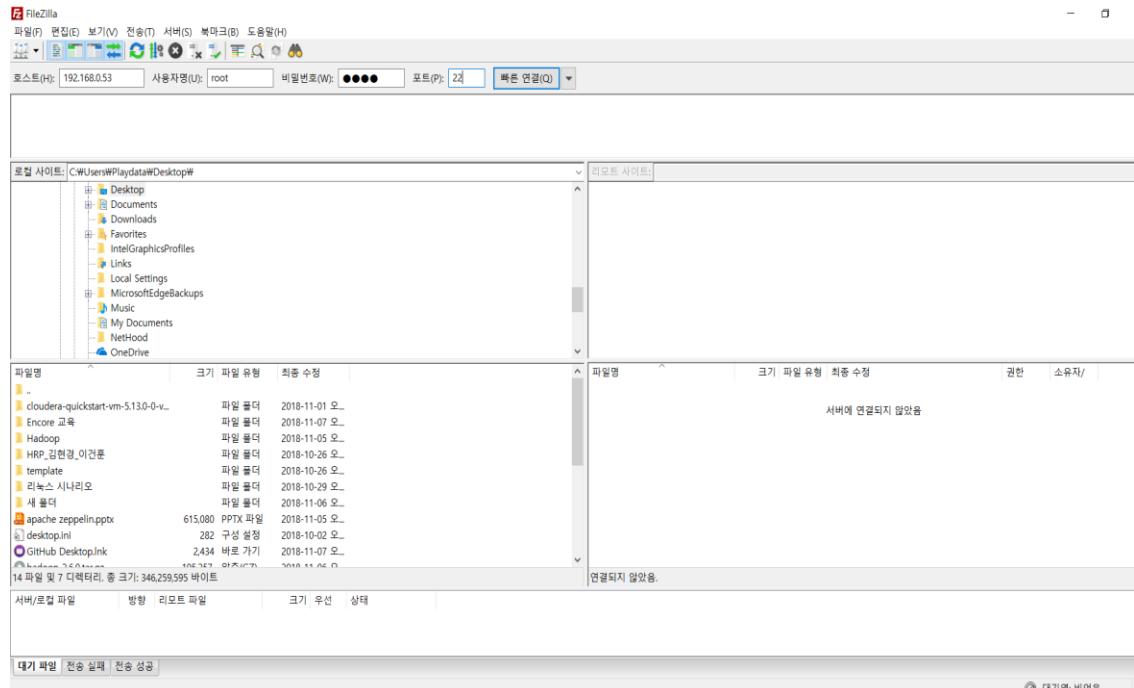
<https://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase7-521261.html>

File

Java SE Development Kit 7u67
Linux x64 (jdk-7u67-linux-x64.tar.gz, 135.78MB)

| Java SE Development Kit 7u67 | | |
|---|--|---|
| You must accept the Oracle Binary Code License Agreement for Java SE to download this software. | | |
| <input type="checkbox"/> Accept License Agreement | <input checked="" type="radio"/> Decline License Agreement | |
| <hr/> | | |
| Product / File Description | File Size | Download |
| Linux x86 | 119.67 MB | jdk-7u67-linux-i586.rpm |
| Linux x86 | 136.94 MB | jdk-7u67-linux-i586.tar.gz |
| Linux x64 | 120.98 MB | jdk-7u67-linux-x64.rpm |
| Linux x64 | 135.78 MB | jdk-7u67-linux-x64.tar.gz |
| Mac OS X x64 | 186.01 MB | jdk-7u67-macosx-x64.dmg |
| Solaris SPARC (SVR4 package) | 138.77 MB | jdk-7u67-solaris-sparc.tar.Z |
| Solaris SPARC | 98.61 MB | jdk-7u67-solaris-sparc.tar.gz |
| Solaris SPARC 64-bit (SVR4 package) | 23.99 MB | jdk-7u67-solaris-sparcv9.tar.Z |
| Solaris SPARC 64-bit | 18.39 MB | jdk-7u67-solaris-sparcv9.tar.gz |
| Solaris x64 (SVR4 package) | 24.74 MB | jdk-7u67-solaris-x64.tar.Z |
| Solaris x64 | 16.35 MB | jdk-7u67-solaris-x64.tar.gz |
| Solaris x86 (SVR4 package) | 139.39 MB | jdk-7u67-solaris-i586.tar.Z |
| Solaris x86 | 95.47 MB | jdk-7u67-solaris-i586.tar.gz |
| Windows x86 | 127.98 MB | jdk-7u67-windows-i586.exe |
| Windows x64 | 129.7 MB | jdk-7u67-windows-x64.exe |

가상머신 실행



Filezilla 실행

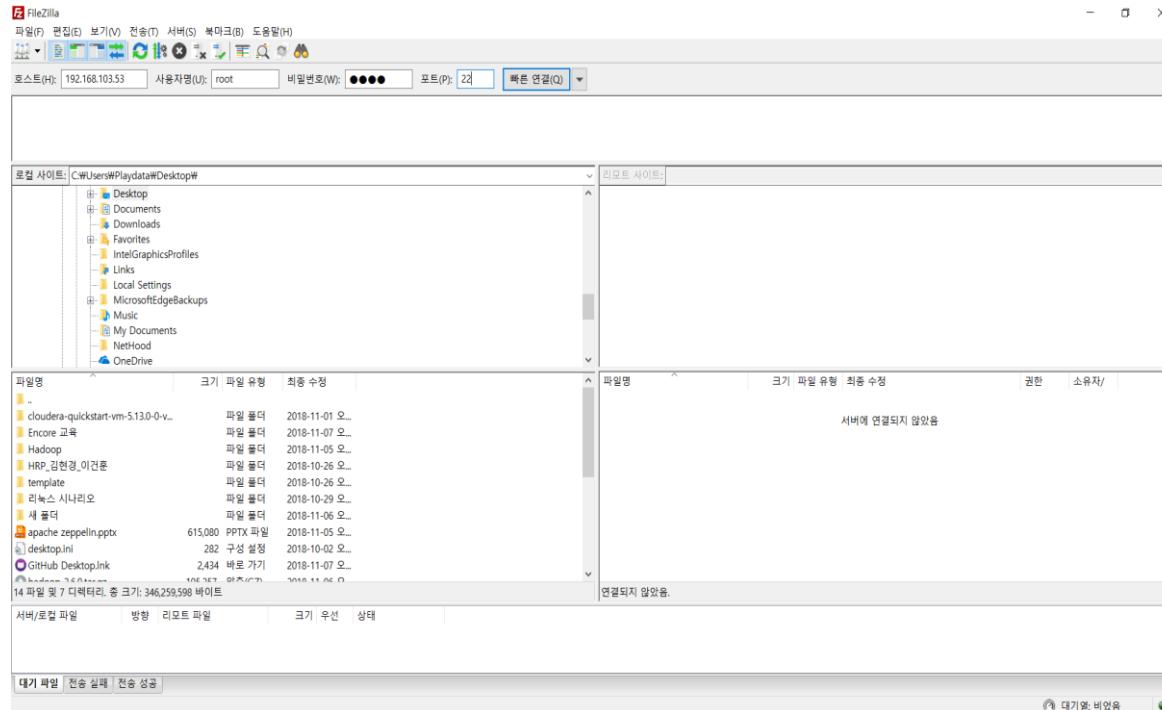
호스트: 고정 ip (ifcfg-enp0s8의 고정 ip)

사용자명 : root

비밀번호 : 사용자명의 비밀번호

포트 : 22

가상머신 실행



Filezilla 실행

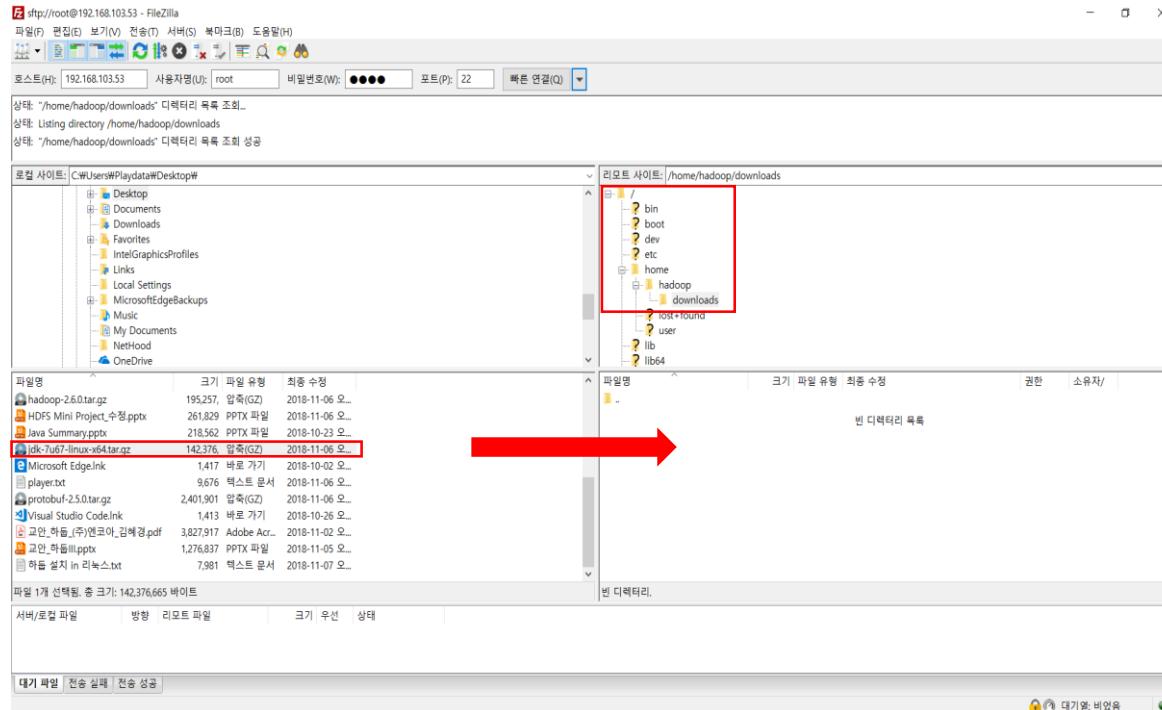
호스트: 고정 ip (ifcfg-enp0s8의 고정 ip)

사용자명 : root

비밀번호 : 사용자명의 비밀번호

포트 : 22

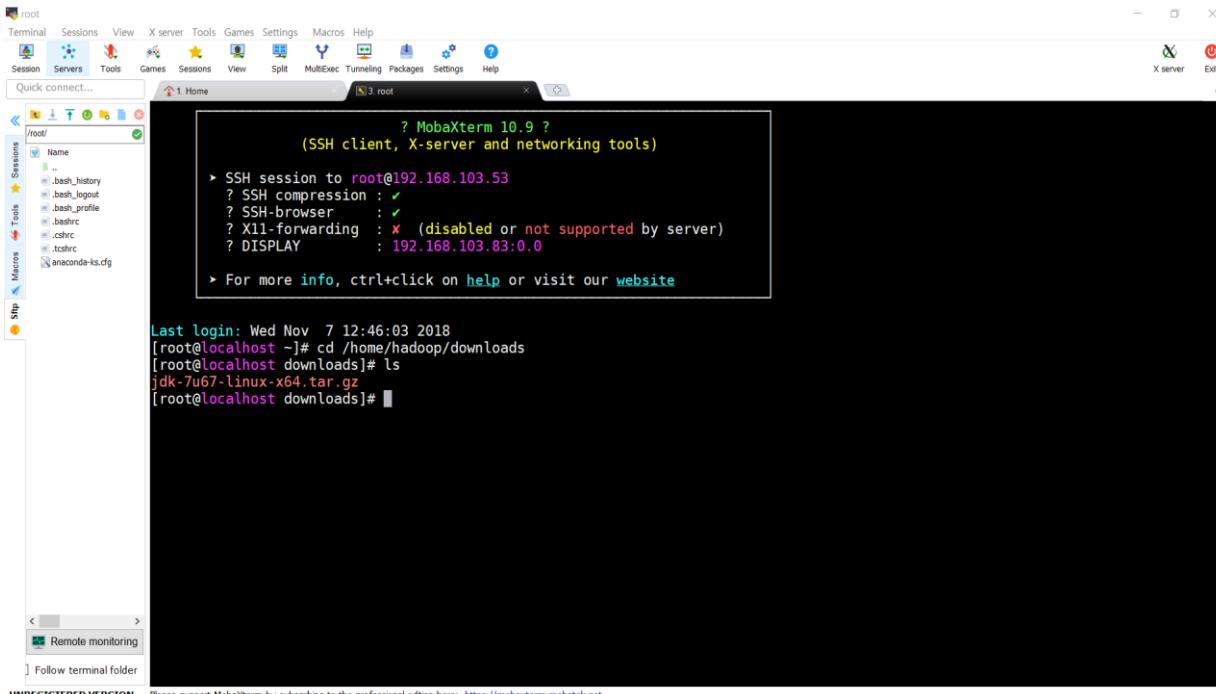
가상머신 실행



Filezilla 실행

오라클 사이트에서 다운로드 받은 jdk-7u67-linux-x64.tar.gz 파일을 가상머신의 /home/hadoop/downloads 디렉토리에 복사

가상머신 실행

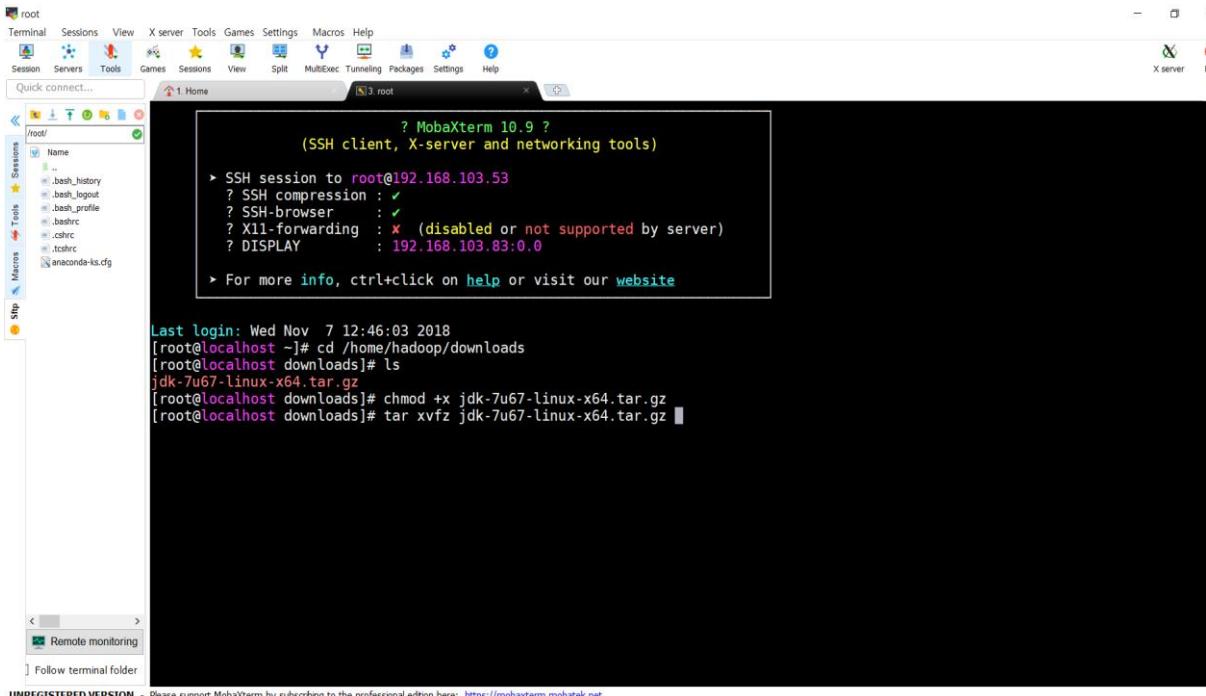


/home/hadoop/downloads 디렉토리에 다운로드 받은 jdk-7u67-linux-x64.tar.gz 파일 확인

[root@localhost downloads]# ls

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

가상머신 실행



/home/hadoop/downloads 디렉토리에 다운로드 받은 jdk-7u67-linux-x64.tar.gz 파일 압축 해제

```
[root@localhost downloads]# chmod +x jdk-7u67-linux-x64.tar.gz
[root@localhost downloads]# tar xvfz jdk-7u67-linux-x64.tar.gz
```

가상머신 실행

MobaXterm 10.9
(SSH client, X-server and networking tools)

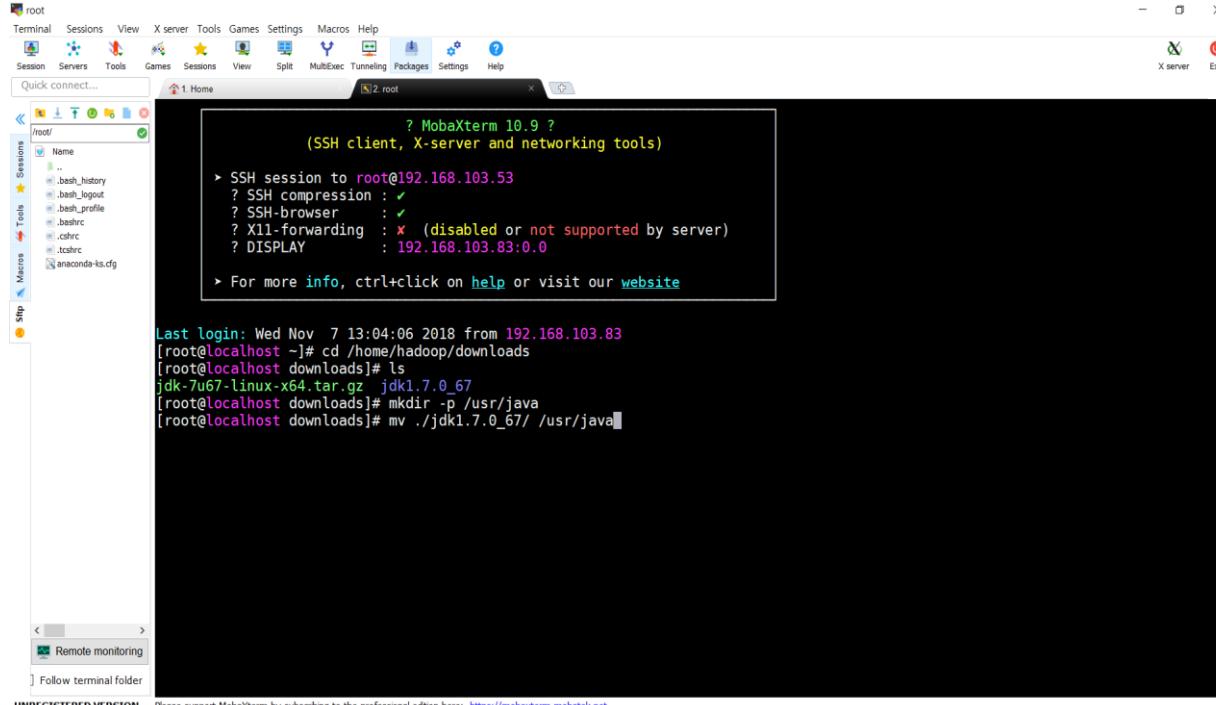
SSH session to root@192.168.103.53
? SSH compression : ✓
? SSH-browser : ✓
? X11-forwarding : ✘ (disabled or not supported by server)
? DISPLAY : 192.168.103.83:0.0

Last login: Wed Nov 7 13:04:06 2018 from 192.168.103.83
[root@localhost ~]# cd /home/hadoop/downloads
[root@localhost downloads]# ls
jdk-7u67-linux-x64.tar.gz jdk1.7.0_67
[root@localhost downloads]# mkdir -p /usr/java
[root@localhost downloads]# mv ./jdk1.7.0_67/ /usr/java

/usr 하위에 java 디렉토리 생성

```
[root@localhost downloads]# mkdir -p /usr/java
```

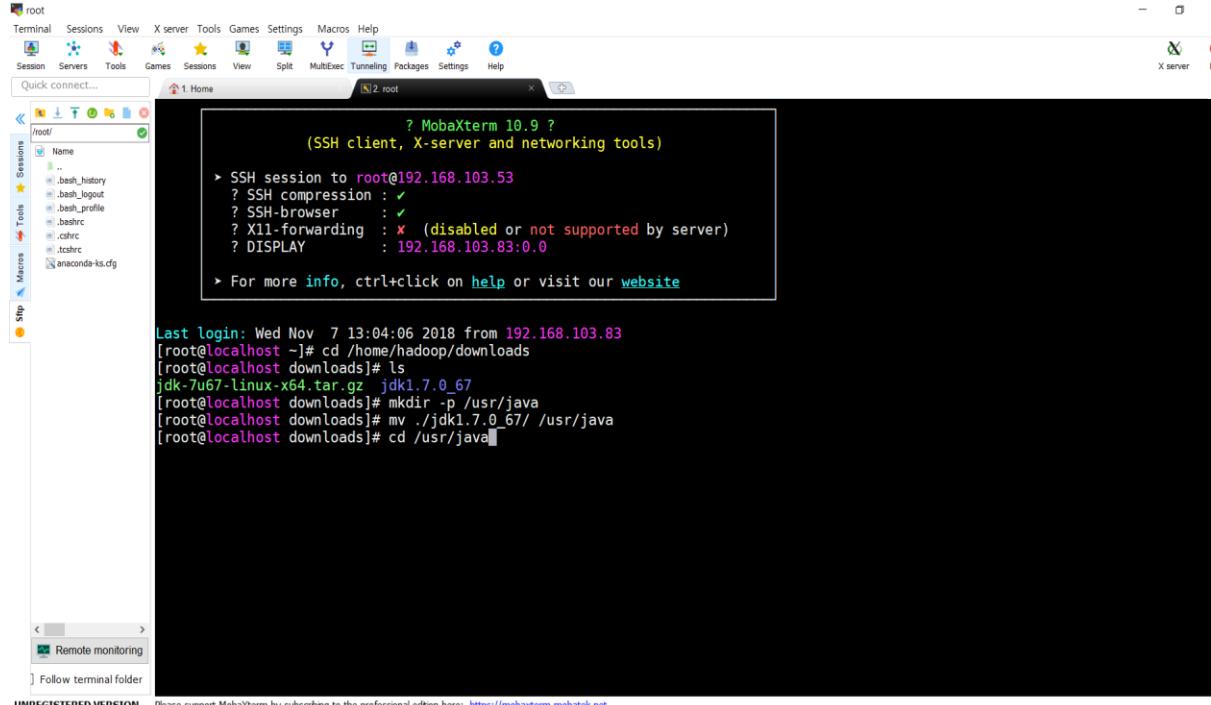
가상머신 실행



압축 해제한 jdk 파일을 /usr/java 디렉토리로 이동

```
[root@localhost downloads]# mv ./jdk1.7.0_67/ /usr/java
```

가상머신 실행



The screenshot shows a MobaXterm window titled 'root' with a terminal session. The terminal output is as follows:

```
? MobaXterm 10.9 ?
(SSH client, X-server and networking tools)

> SSH session to root@192.168.103.53
? SSH compression : ✓
? SSH-browser : ✓
? X11-forwarding : ✘ (disabled or not supported by server)
? DISPLAY : 192.168.103.83:0.0

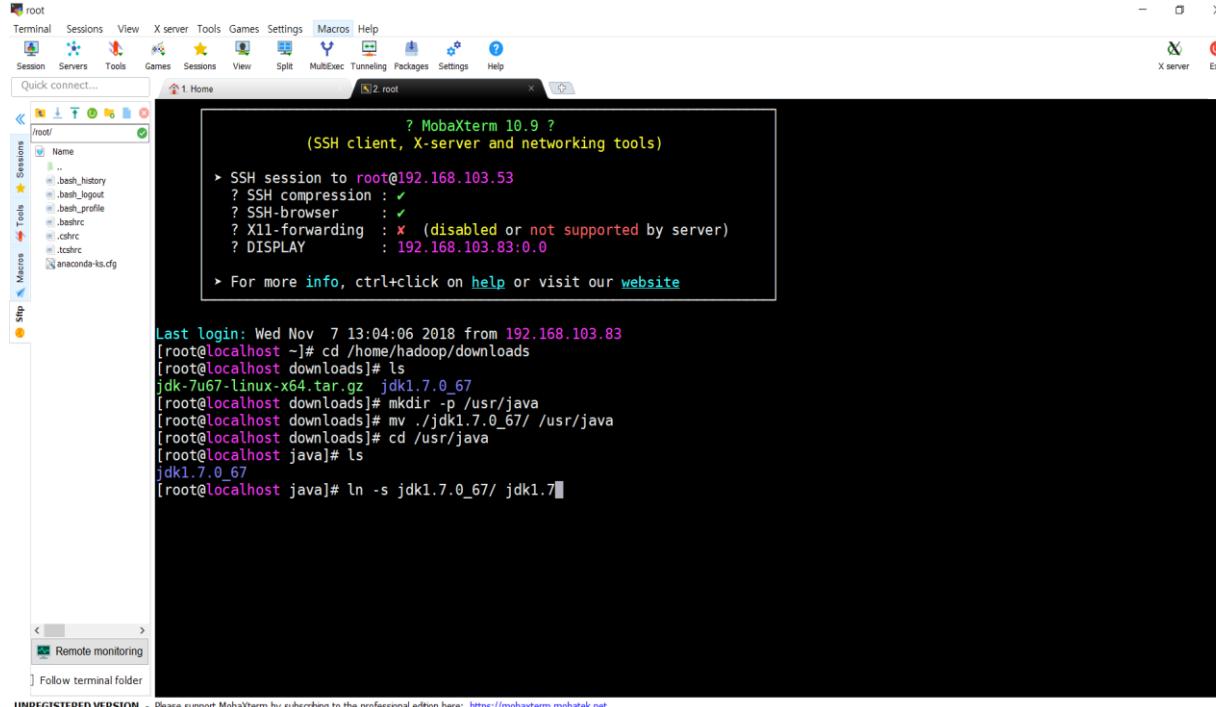
> For more info, ctrl+click on help or visit our website

Last login: Wed Nov  7 13:04:06 2018 from 192.168.103.83
[root@localhost ~]# cd /home/hadoop/downloads
[root@localhost downloads]# ls
jdk-7u67-linux-x64.tar.gz jdk1.7.0_67
[root@localhost downloads]# mkdir -p /usr/java
[root@localhost downloads]# mv ./jdk1.7.0_67/ /usr/java
[root@localhost downloads]# cd /usr/java
```

/usr/java 디렉토리로 이동

```
[root@localhost downloads]# cd /usr/java
```

가상머신 실행



The screenshot shows a MobaXterm window with a terminal session titled 'root'. The session details indicate it's connected via SSH to root@192.168.103.53. The terminal window displays the following command sequence:

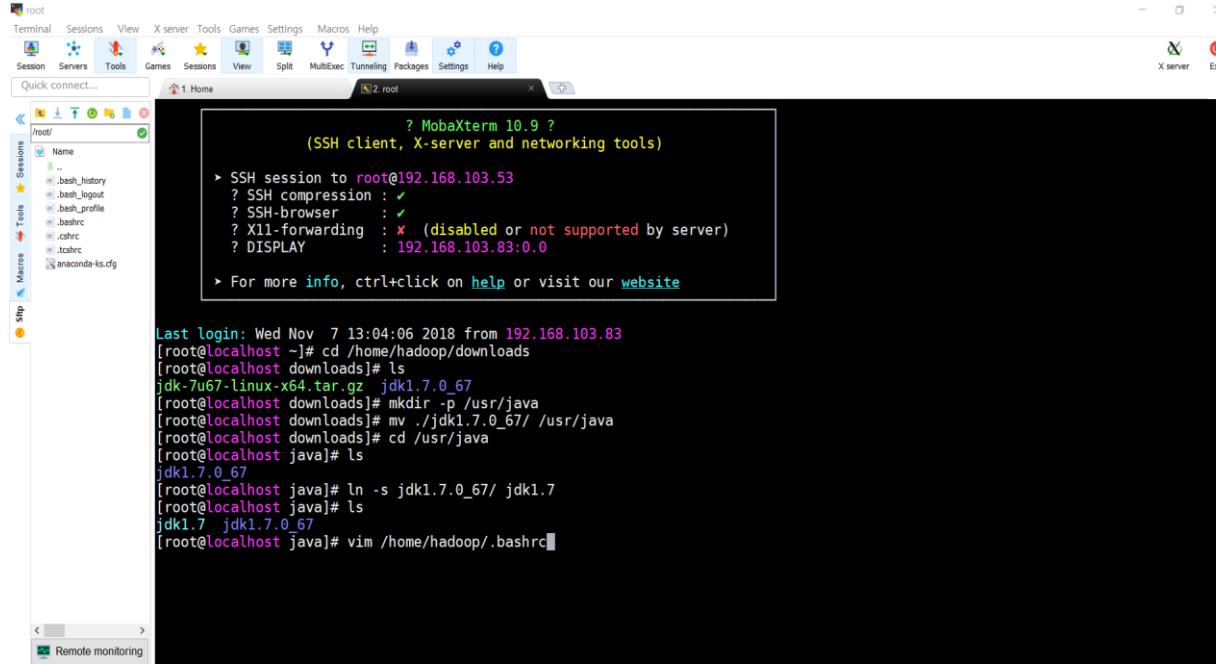
```
Last login: Wed Nov  7 13:04:06 2018 from 192.168.103.83
[root@localhost ~]# cd /home/hadoop/downloads
[root@localhost downloads]# ls
jdk-7u67-linux-x64.tar.gz jdk1.7.0_67
[root@localhost downloads]# mkdir -p /usr/java
[root@localhost downloads]# mv ./jdk1.7.0_67/ /usr/java
[root@localhost downloads]# cd /usr/java
[root@localhost java]# ls
jdk1.7.0_67
[root@localhost java]# ln -s jdk1.7.0_67/ jdk1.7
```

At the bottom left of the terminal window, there is a note: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

jdk1.7.0_67의 명칭 변경해서 새로 생성

```
[root@localhost java]# ln -s jdk1.7.0_67/ jdk1.7
```

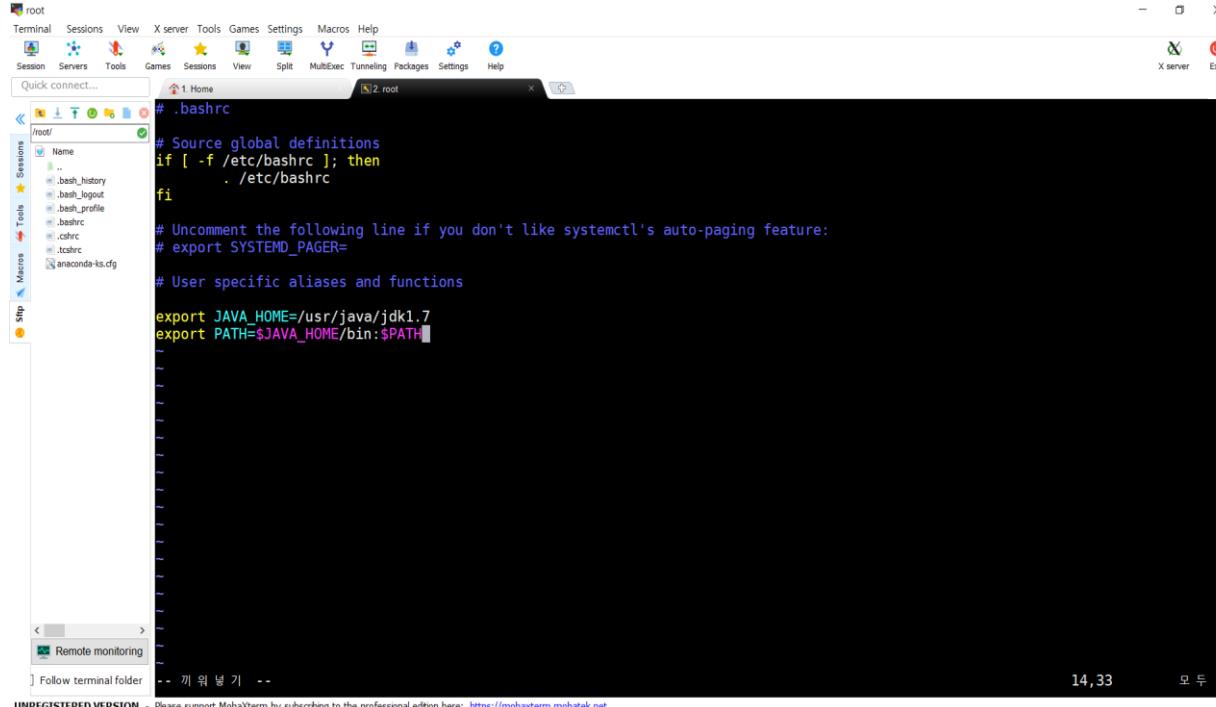
가상머신 실행



/home/hadoop/.bashrc를 vim으로 실행해 편집

```
[root@localhost java]# vim /home/hadoop/.bashrc
```

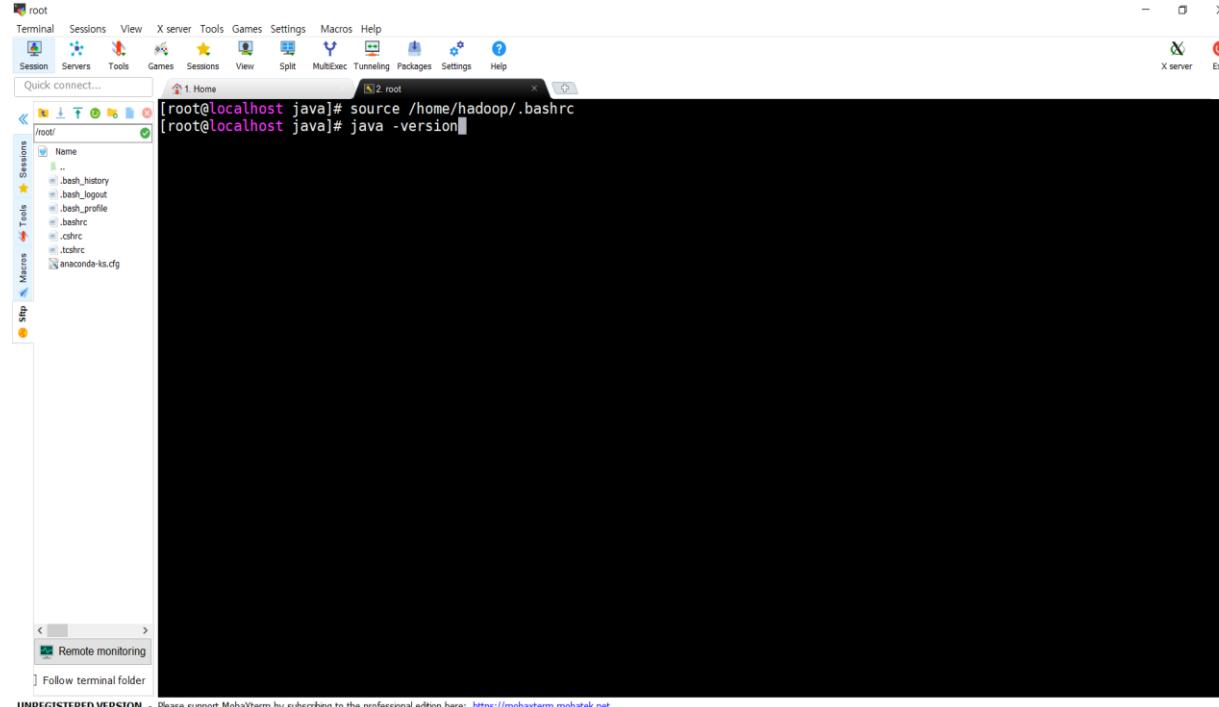
가상머신 실행



/home/hadoop/.bashrc를 vim으로 실행해 편집

```
export JAVA_HOME=/usr/java/jdk1.7  
export PATH=$JAVA_HOME/bin:$PATH
```

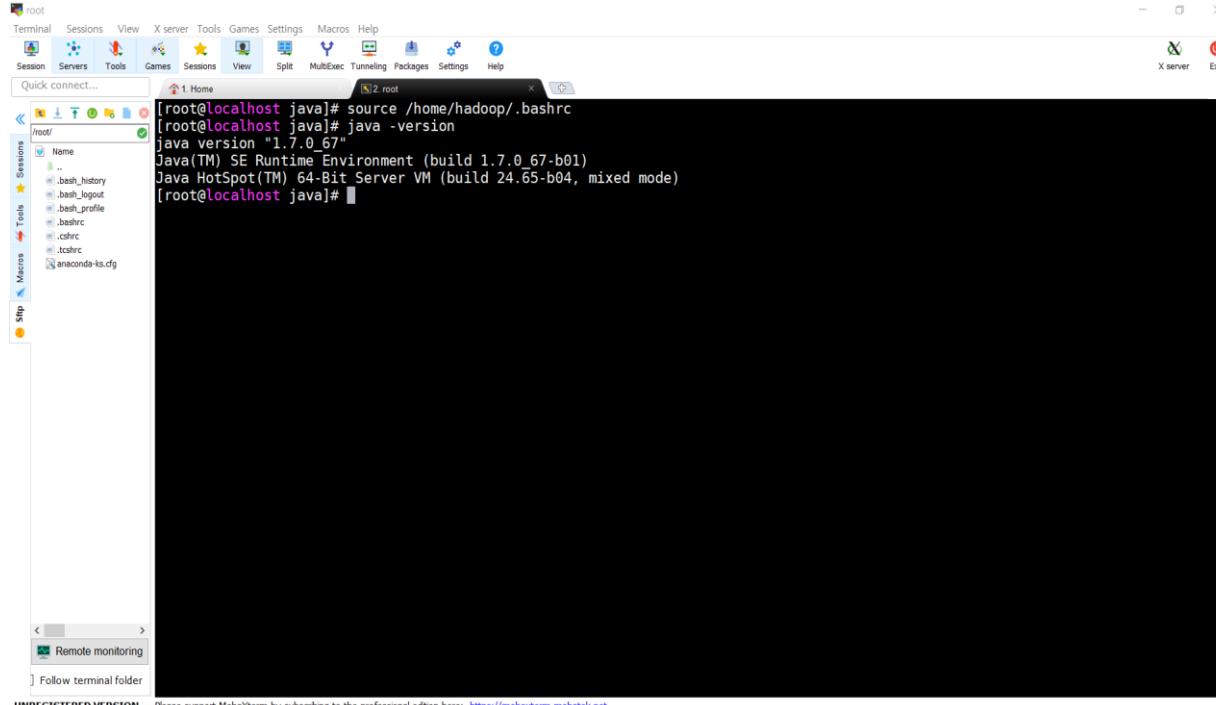
가상머신 실행



/home/hadoop/.bashrc 적용 및 java 버전 확인

```
[root@localhost java]# source /home/hadoop/.bashrc
[root@localhost java]# java -version
```

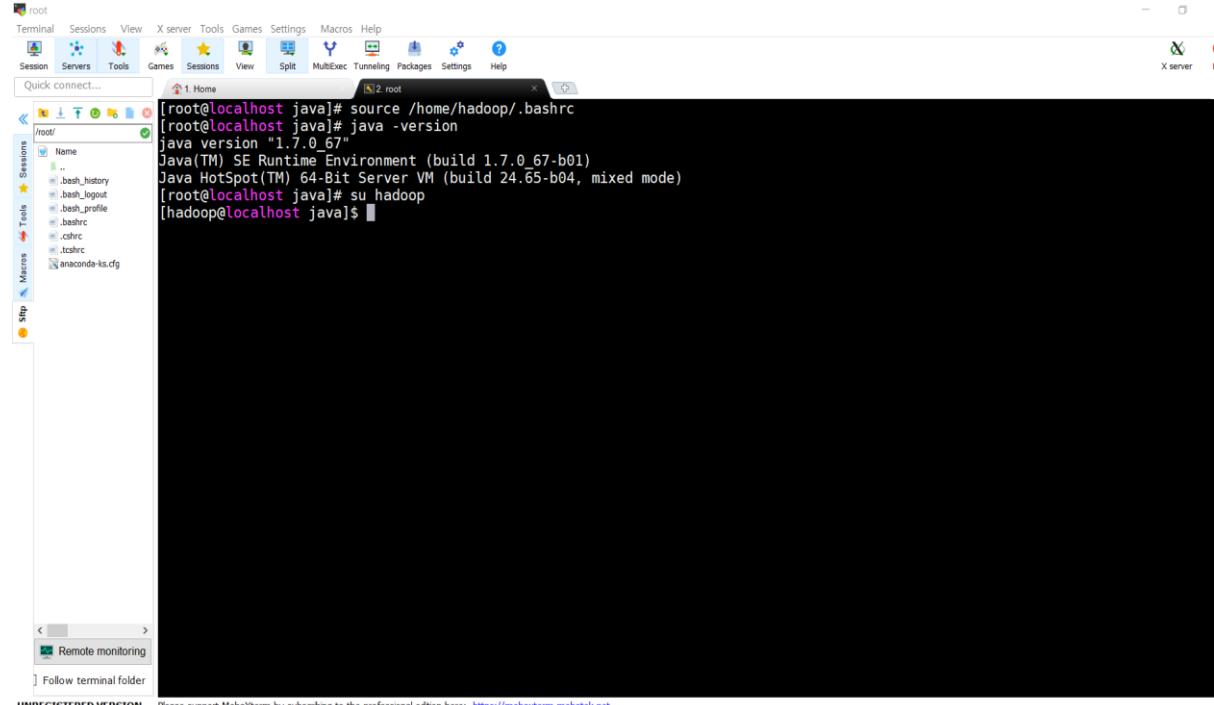
가상머신 실행



/home/hadoop/.bashrc 적용 및 java 버전 확인

```
[root@localhost java]# source /home/hadoop/.bashrc
[root@localhost java]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
```

가상머신 실행



A screenshot of the MobaXterm application interface. The main window shows a terminal session for user 'root' on 'localhost'. The terminal window title is 'root' and the tab title is 'root'. The terminal content shows the following command execution:

```
[root@localhost java]# source /home/hadoop/.bashrc
[root@localhost java]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
[root@localhost java]# su hadoop
[hadoop@localhost java]$
```

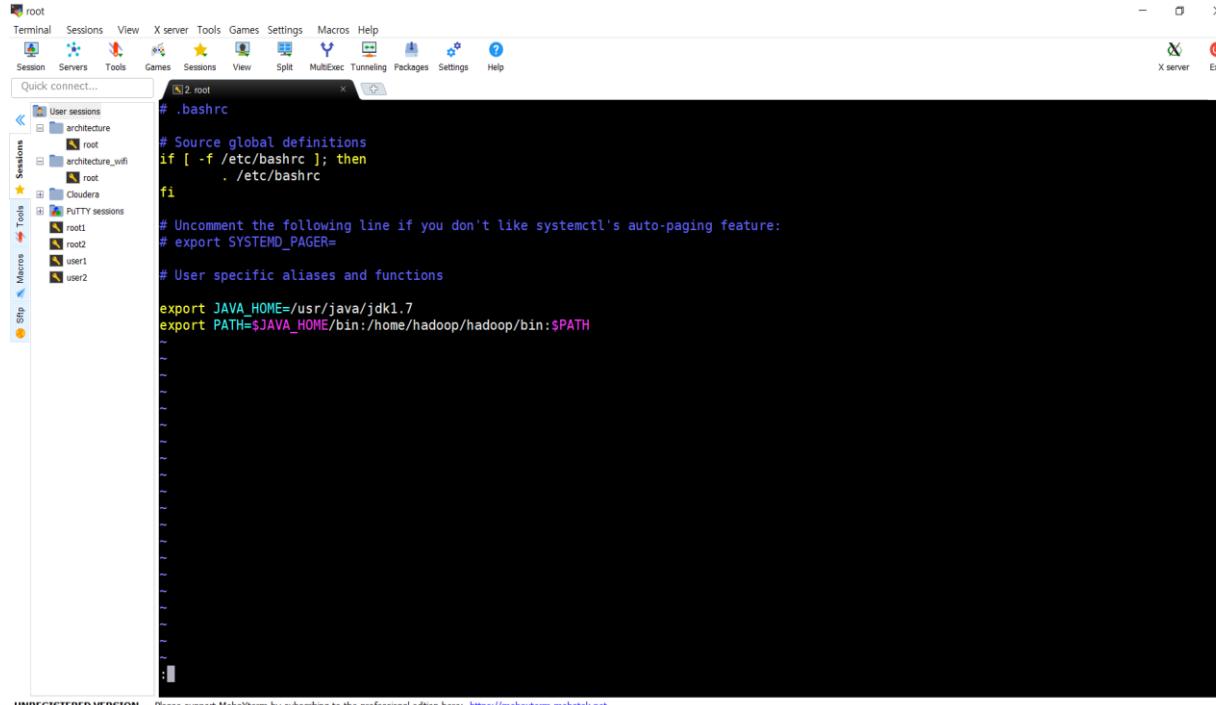
The MobaXterm interface includes a menu bar with options like Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, and Help. On the left, there's a sidebar with Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, and Help buttons. Below the terminal window, there are sections for Remote monitoring and Follow terminal folder.

hadoop 계정으로 사용자 변경

```
[root@localhost java]# su hadoop
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

가상머신 실행



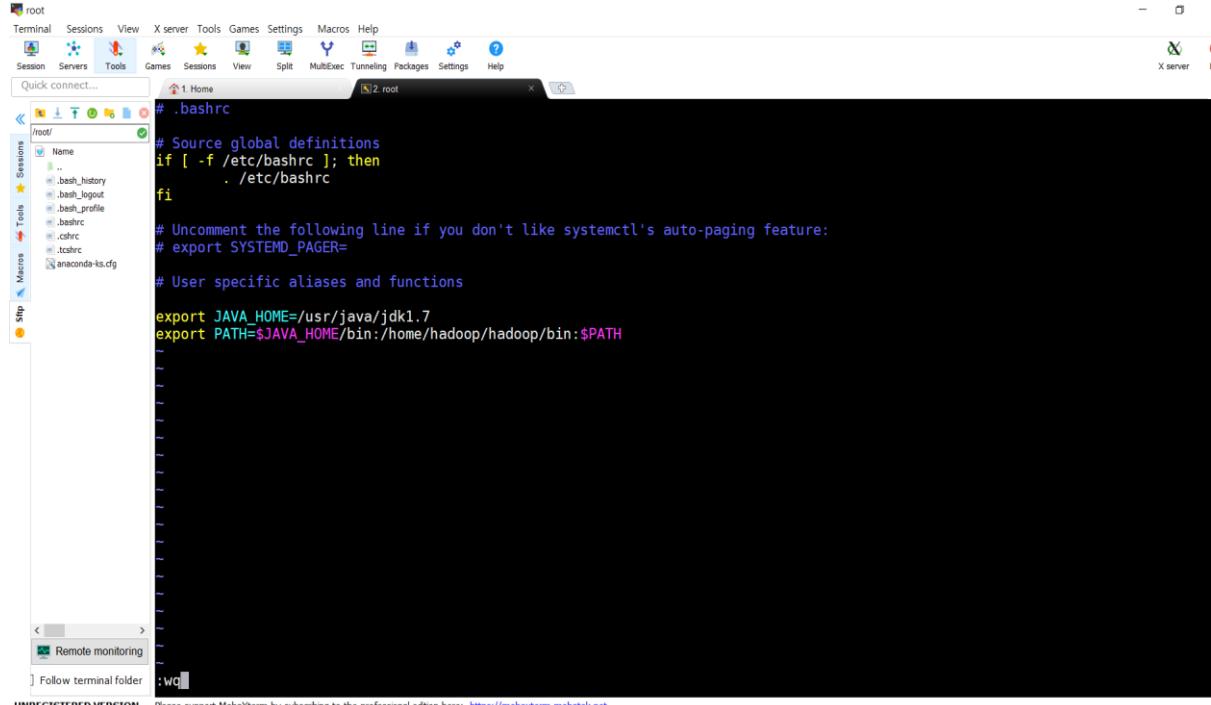
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

/home/hadoop/.bashrc 파일 vim으로 편집

```
[hadoop@localhost java]$ vim /home/hadoop/.bashrc

export PATH=$JAVA_HOME/bin:$PATH를
export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop/bin:$PATH
```

가상머신 실행



A screenshot of the MobaXterm interface. On the left, there's a sidebar with tabs for Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, and Help. Under Sessions, there are icons for Session, Servers, and Tools. The main window shows a terminal session for user 'root' at '/root'. The terminal title is '# .bashrc'. The content of the file is displayed in blue font:

```
# .bashrc
#
# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions

export JAVA_HOME=/usr/java/jdk1.7
export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop/bin:$PATH
```

The bottom of the terminal window shows the command ':wq' entered in the input field.

/home/hadoop/.bashrc 파일 vim으로 편집

```
[hadoop@localhost java]$ vim /home/hadoop/.bashrc
export PATH=$JAVA_HOME/bin:$PATH를
export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop/bin:$PATH
```

가상머신 실행

Protocol Buffers v3.0.0-alpha-1

v3.0.0-alpha-1 · 8d5d7cc · Pre-release · xfyjwrf · 11 Dec 2014 · 5025 commits to master since this release

Assets

| File | Size |
|------------------------------------|---------|
| protobuf-cpp-3.0.0-alpha-1.tar.gz | 2.26 MB |
| protobuf-cpp-3.0.0-alpha-1.zip | 2.82 MB |
| protobuf-java-3.0.0-alpha-1.tar.gz | 2.54 MB |
| protobuf-java-3.0.0-alpha-1.zip | 3.23 MB |
| protoc-3.0.0-alpha-1-win32.zip | 808 KB |

Source code (zip)

Source code (tar.gz)

Version 3.0.0-alpha-1 (C++/Java)

v2.5.0 · 774d630 · xfyjwrf · 25 Mar 2015 · 5815 commits to master since this release

Protocol Buffers v2.5.0

Assets

| File | Size |
|------------------------|---------|
| protobuf-2.5.0.tar.bz2 | 1.78 MB |
| protobuf-2.5.0.tar.gz | 2.29 MB |
| protobuf-2.5.0.zip | 2.91 MB |
| protoc-2.5.0-win32.zip | 638 KB |

Source code (zip)

Source code (tar.gz)

Protocol Buffer 2.5.0 버전 다운로드 받은 후
"Filezilla"를 통해 /home/hadoop/downloads에 업로드

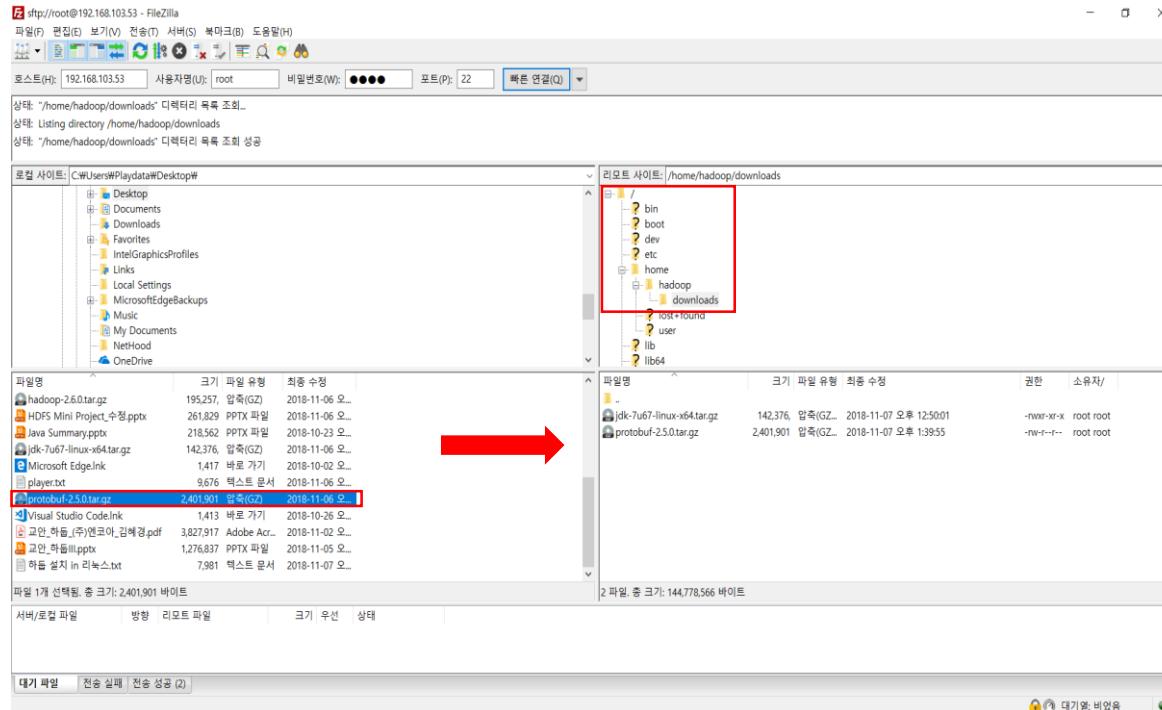
Url

<https://github.com/protocolbuffers/protobuf/releases?after=v3.0.0-alpha-2>

File

Protocol Buffers v2.5.0
protobuf-2.5.0.tar.gz, 2.29MB

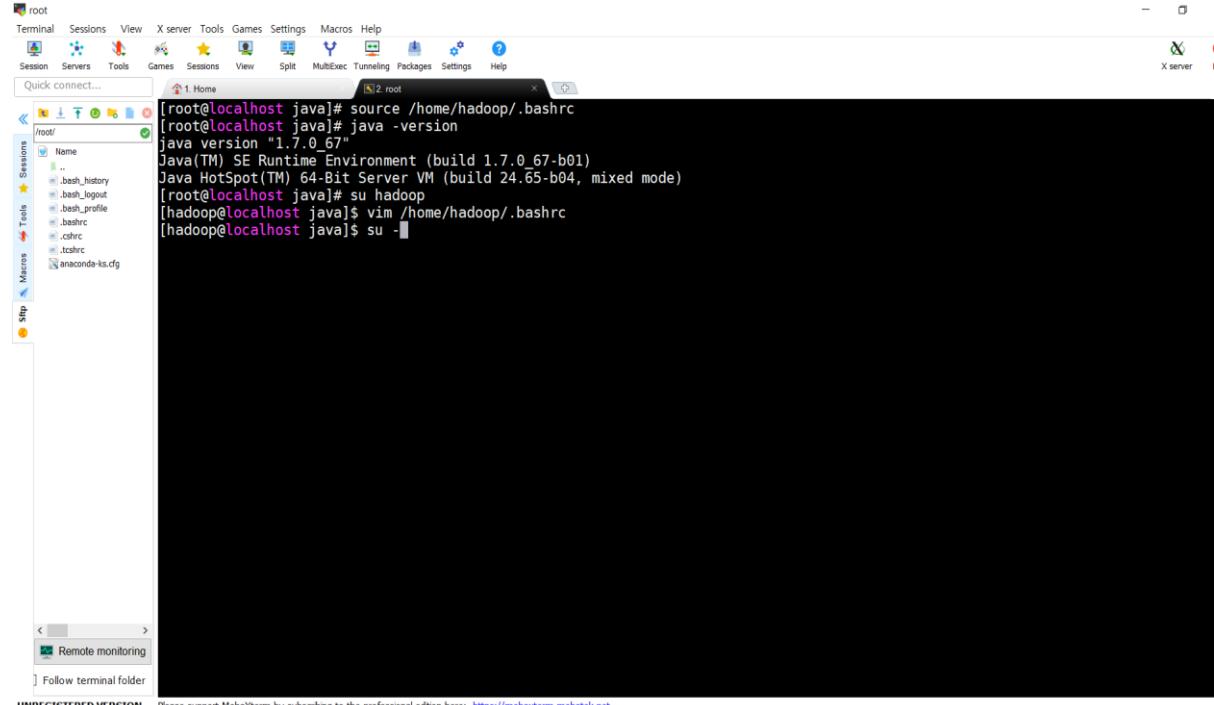
가상머신 실행



Filezilla 실행

Protocol buffers github 사이트에서 다운로드 받은 protobuf-2.5.0.tar.gz 파일을 가상머신의 /home/hadoop/downloads 디렉토리에 복사

가상머신 실행



A screenshot of the MobaXterm application interface. The title bar says "root". The menu bar includes "Terminal", "Sessions", "View", "X server", "Tools", "Games", "Settings", "Macros", and "Help". The toolbar has icons for Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, and Help. The main window shows a terminal session as root. The command history shows:

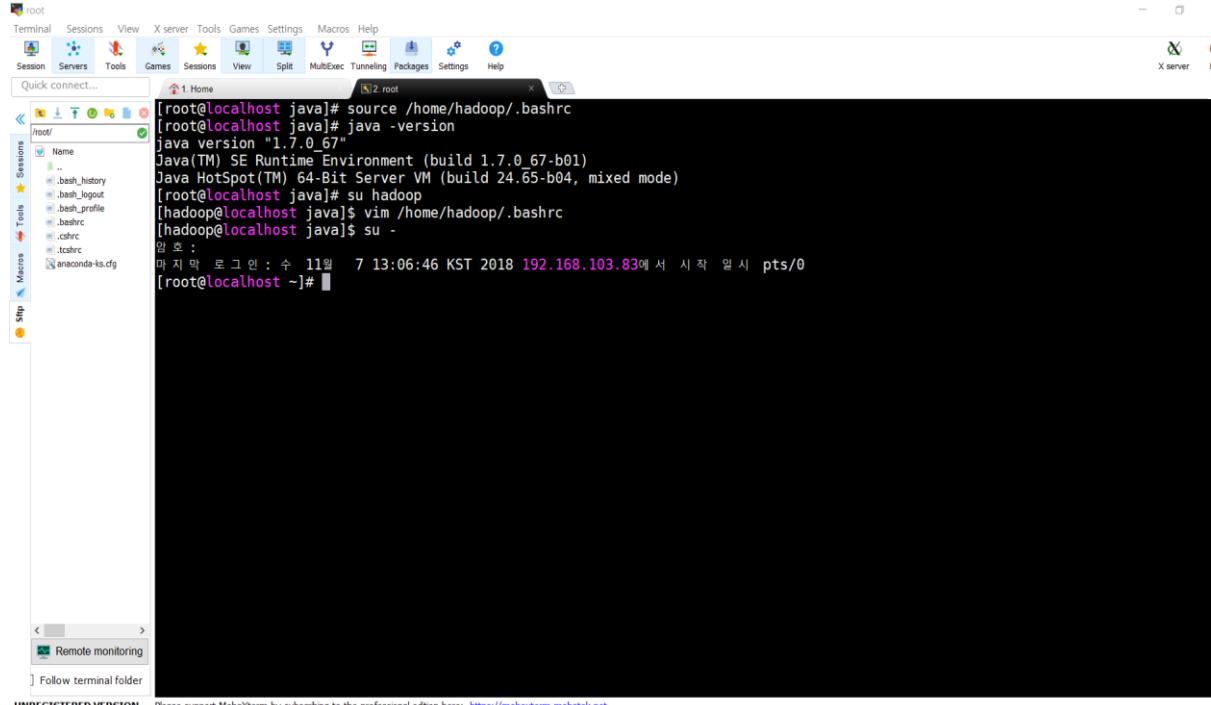
```
[root@localhost java]# source /home/hadoop/.bashrc
[root@localhost java]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
[root@localhost java]# su hadoop
[hadoop@localhost java]$ vim /home/hadoop/.bashrc
[hadoop@localhost java]$ su -
```

The left sidebar shows sessions named "root" and "hadoop". The bottom status bar indicates "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

root 계정으로 사용자 변경

```
[hadoop@localhost java]$ su -
```

가상머신 실행



A screenshot of the MobaXterm application interface. The title bar says "root". The menu bar includes Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, and Help. The toolbar has icons for Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, and Help. The main window shows a terminal session as root. The command history shows:

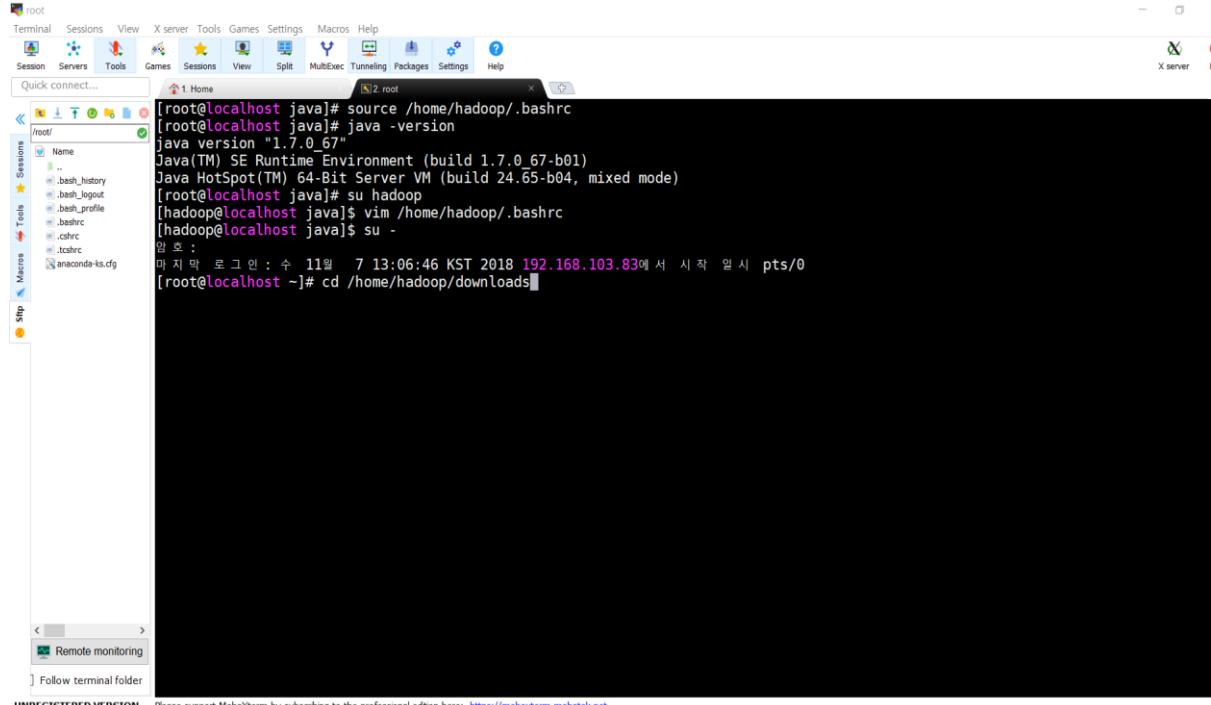
```
[root@localhost java]# source /home/hadoop/.bashrc
[root@localhost java]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
[root@localhost java]# su hadoop
[hadoop@localhost java]$ vim /home/hadoop/.bashrc
[hadoop@localhost java]$ su -
암호 :
마지막 로그인: 수 11월 7 13:06:46 KST 2018 192.168.103.83에서 시작 일시 pts/0
[root@localhost ~]#
```

The sidebar on the left shows sessions named "root" and "hadoop". The bottom status bar indicates "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

root 계정으로 사용자 변경

```
[hadoop@localhost java]$ su -
```

가상머신 실행

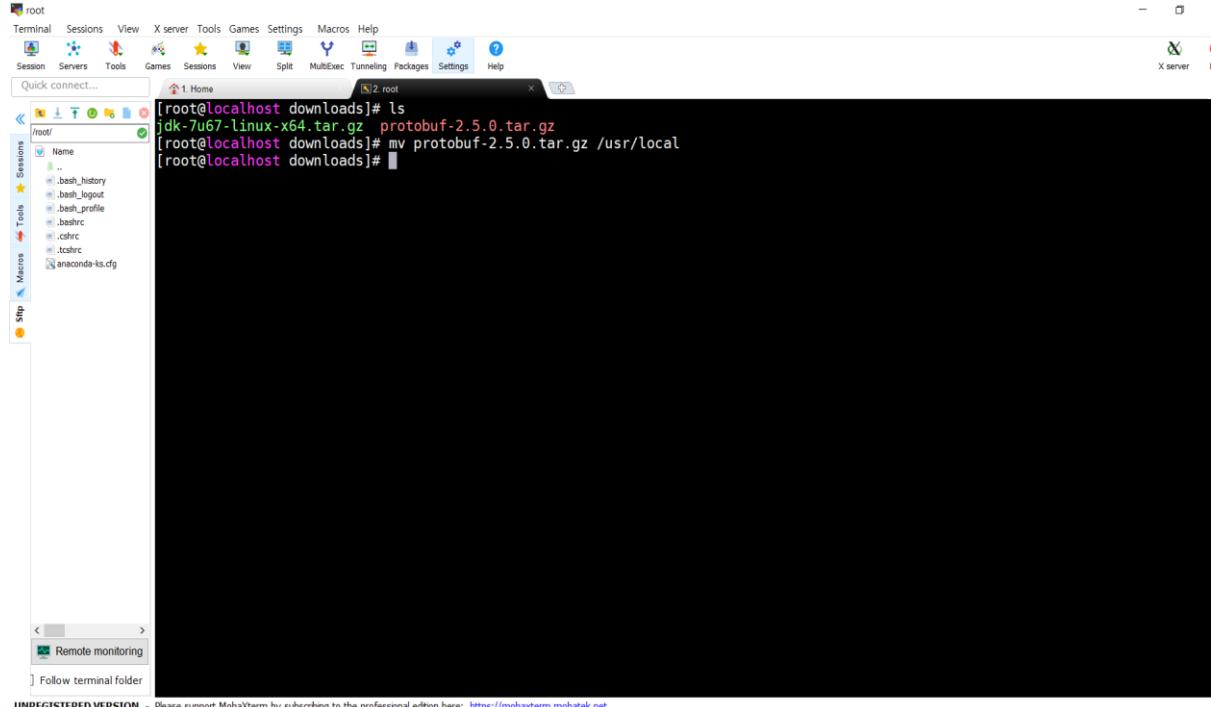


```
[root@localhost java]# source /home/hadoop/.bashrc
[root@localhost java]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
[root@localhost java]# su hadoop
[hadoop@localhost java]$ vim /home/hadoop/.bashrc
[hadoop@localhost java]$ su -
암호 :
마지막 로그인 : 수 11월 7 13:06:46 KST 2018 192.168.103.83에서 시작 일시 pts/0
[root@localhost ~]# cd /home/hadoop/downloads
```

/home/hadoop/downloads 디렉토리로 이동

```
[root@localhost ~]# cd /home/hadoop/downloads
```

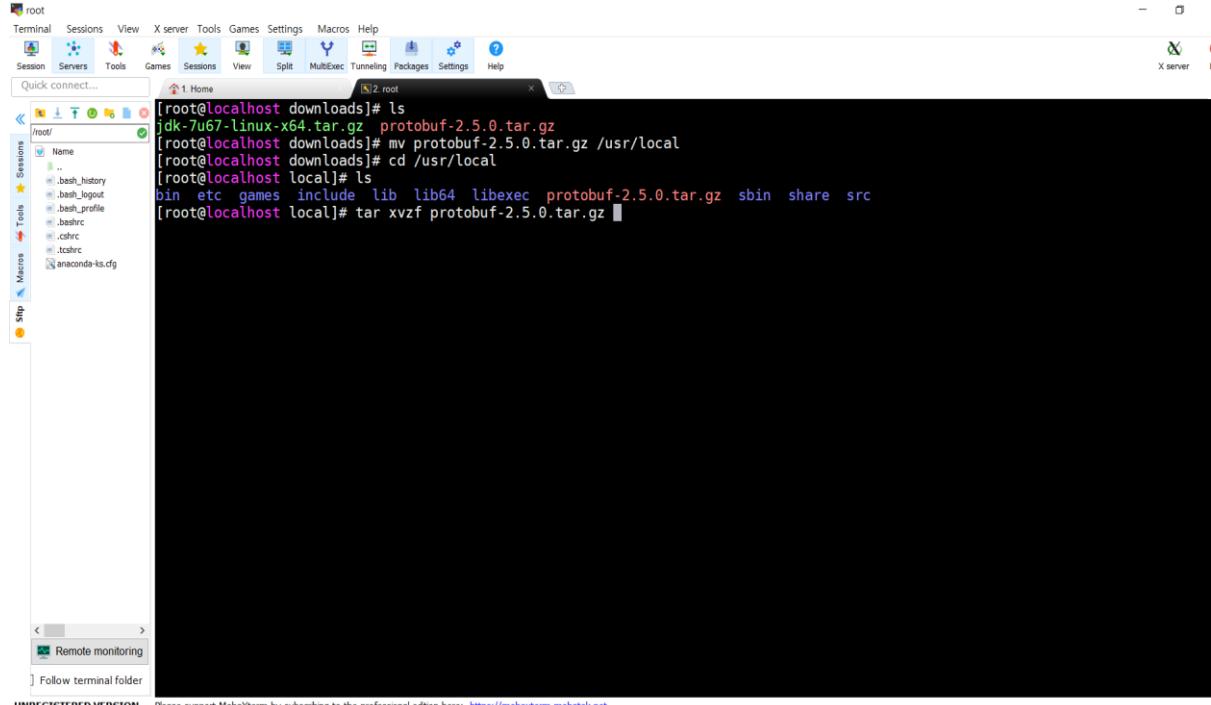
가상머신 실행



downloads 디렉토리에 위치한 protobuf-2.5.0.tar.gz 파일을 /usr/local로 이동

```
[root@localhost ~]# mv protobuf-2.5.0.tar.gz /usr/local
```

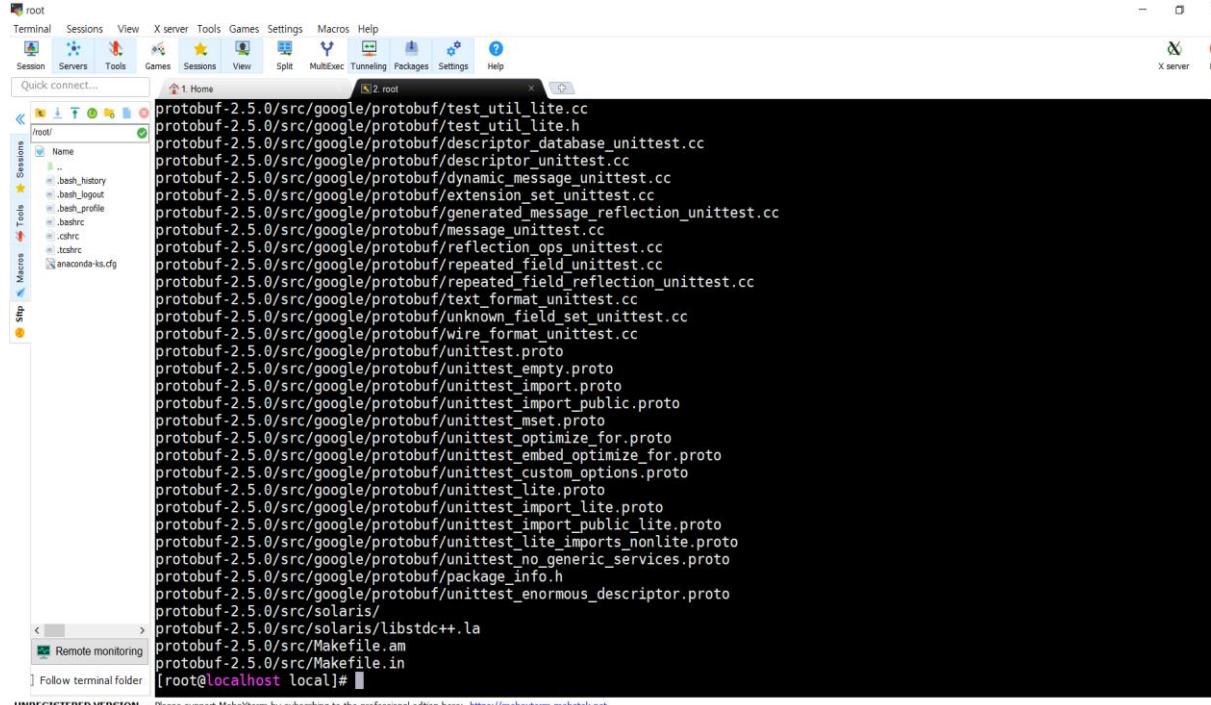
가상머신 실행



/usr/local 디렉토리에서 protobuf-2.5.0.tar.gz 파일 압축 해제

```
[root@localhost ~]# cd /usr/local
[root@localhost local]# tar xvzf protobuf-2.5.0.tar.gz
```

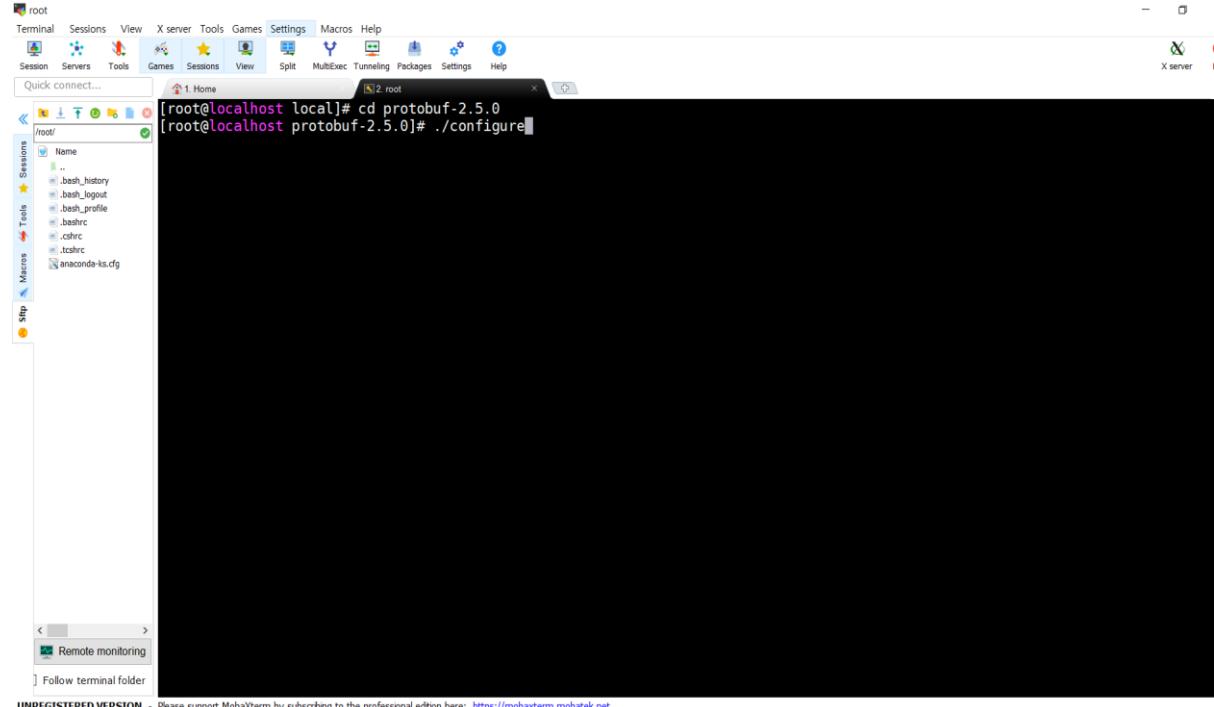
가상머신 실행



/usr/local 디렉토리에서 protobuf-2.5.0.tar.gz 파일 압축 해제

```
[root@localhost ~]# cd /usr/local  
[root@localhost local]# tar xvzf protobuf-2.5.0.tar.gz
```

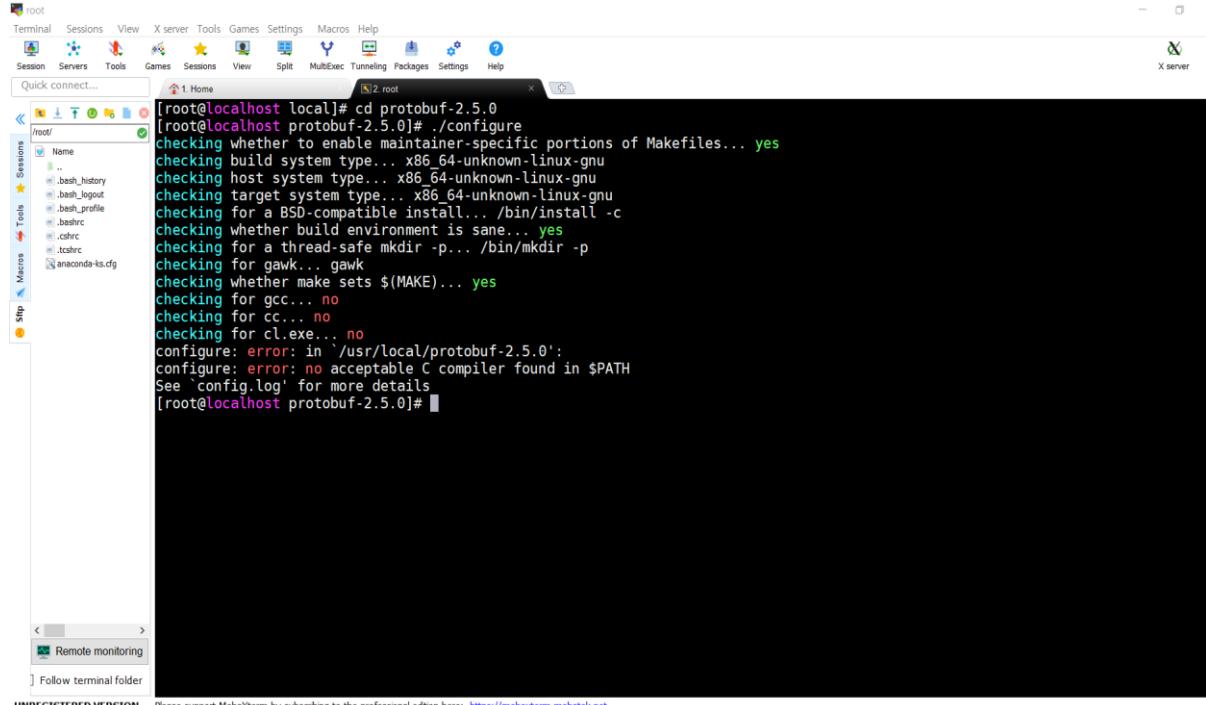
가상머신 실행



Protobuf-2.5.0 디렉토리로 이동 후 configure 파일 실행

```
[root@localhost local]# cd protobuf-2.5.0  
[root@localhost protobuf-2.5.0]# ./configure
```

가상머신 실행



The screenshot shows a MobaXterm terminal window titled 'root' with the following session details:

- Terminal: Sessions, View, X server, Tools, Games, Settings, Macros, Help
- Session: Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, Help
- Tools: Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, Help
- Macros: Step

The terminal window displays the following command and its output:

```
[root@localhost local]# cd protobuf-2.5.0
[root@localhost protobuf-2.5.0]# ./configure
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking target system type... x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... no
checking for cc... no
checking for cl.exe... no
configure: error: in '/usr/local/protobuf-2.5.0':
configure: error: no acceptable C compiler found in $PATH
See `config.log' for more details
[root@localhost protobuf-2.5.0]#
```

At the bottom left of the terminal window, there is a note: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

Protobuf-2.5.0 디렉토리로 이동 후 configure 파일 실행
오류 발생!!!!!!

가상머신 실행

The screenshot shows a MobaXterm terminal window titled 'root' with session '1. Home'. The terminal displays the following command and its output:

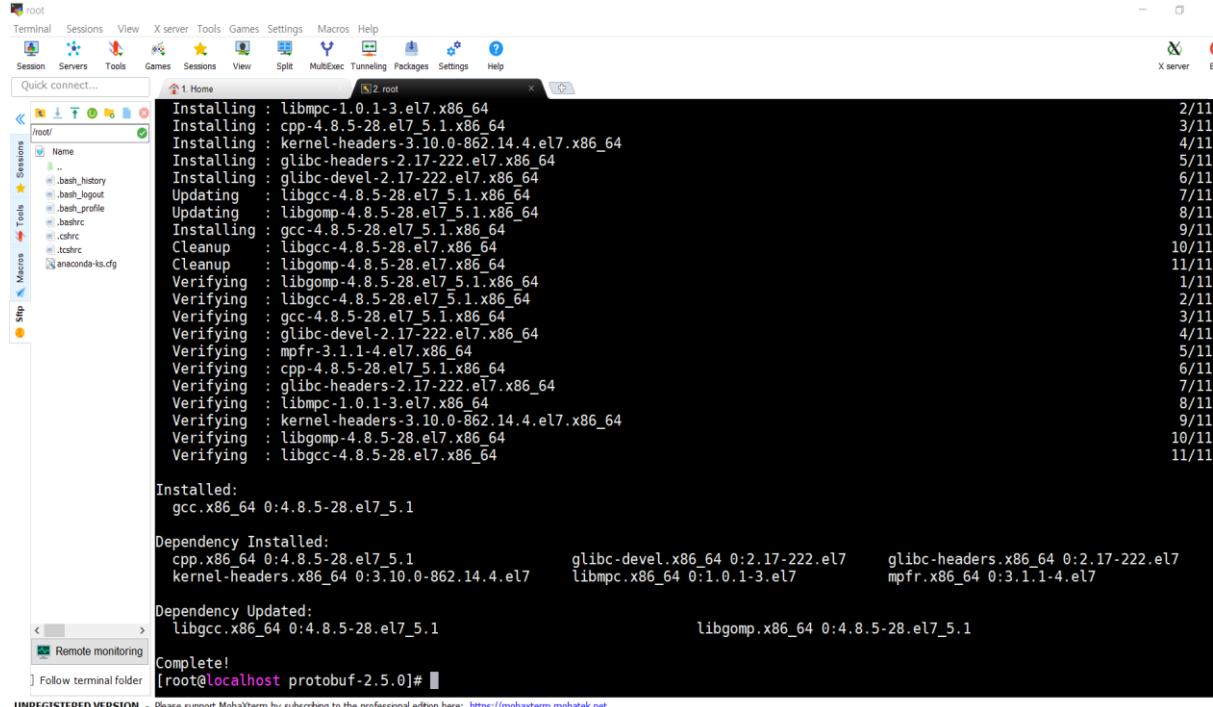
```
[root@localhost local]# cd protobuf-2.5.0
[root@localhost protobuf-2.5.0]# ./configure
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking target system type... x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... no
checking for cc... no
checking for cl.exe... no
configure: error: in '/usr/local/protobuf-2.5.0':
configure: error: no acceptable C compiler found in $PATH
See `config.log' for more details
[root@localhost protobuf-2.5.0]# rpm -qa|grep gcc
libgcc-4.8.5-28.el7.x86_64
[root@localhost protobuf-2.5.0]# yum -y install gcc
```

The terminal interface includes a menu bar (Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, Help), a toolbar, and various session management buttons. The bottom of the window shows the MobaXterm footer with 'UNREGISTERED VERSION' and a link to the professional edition.

Protobuf-2.5.0 디렉토리로 이동 후 configure 파일 실행
오류 발생!!!!!! 해결하기 1

[root@localhost protobuf-2.5.0]# rpm -qa | grep gcc (컴파일러 설치 유무 확인)
[root@localhost protobuf-2.5.0]# yum -y install gcc (gcc 설치)

가상머신 실행



MobaXterm terminal window showing the output of an rpm install command:

```
root@localhost ~# rpm -ivh gcc-4.8.5-28.el7.x86_64
Installing : libmpc-1.0.1-3.el7.x86_64
Installing : cpp-4.8.5-28.el7.5.1.x86_64
Installing : kernel-headers-3.10.0-862.14.4.el7.x86_64
Installing : glibc-headers-2.17-222.el7.x86_64
Installing : glibc-devel-2.17-222.el7.x86_64
Updating   : libgcc-4.8.5-28.el7.5.1.x86_64
Updating   : libgomp-4.8.5-28.el7.5.1.x86_64
Updating   : gcc-4.8.5-28.el7.5.1.x86_64
Cleanup    : libgcc-4.8.5-28.el7.x86_64
Cleanup    : libgomp-4.8.5-28.el7.x86_64
Verifying  : libgomp-4.8.5-28.el7.5.1.x86_64
Verifying  : libgcc-4.8.5-28.el7.5.1.x86_64
Verifying  : gcc-4.8.5-28.el7.5.1.x86_64
Verifying  : glibc-devel-2.17-222.el7.x86_64
Verifying  : mpfr-3.1.1-4.el7.x86_64
Verifying  : cpp-4.8.5-28.el7.5.1.x86_64
Verifying  : glibc-headers-2.17-222.el7.x86_64
Verifying  : libmpc-1.0.1-3.el7.x86_64
Verifying  : kernel-headers-3.10.0-862.14.4.el7.x86_64
Verifying  : libgomp-4.8.5-28.el7.x86_64
Verifying  : libgcc-4.8.5-28.el7.x86_64

Installed:
  gcc.x86_64 0:4.8.5-28.el7_5.1

Dependency Installed:
  cpp.x86_64 0:4.8.5-28.el7_5.1           glibc-devel.x86_64 0:2.17-222.el7      glibc-headers.x86_64 0:2.17-222.el7
  kernel-headers.x86_64 0:3.10.0-862.14.4.el7 libmpc.x86_64 0:1.0.1-3.el7          mpfr.x86_64 0:3.1.1-4.el7

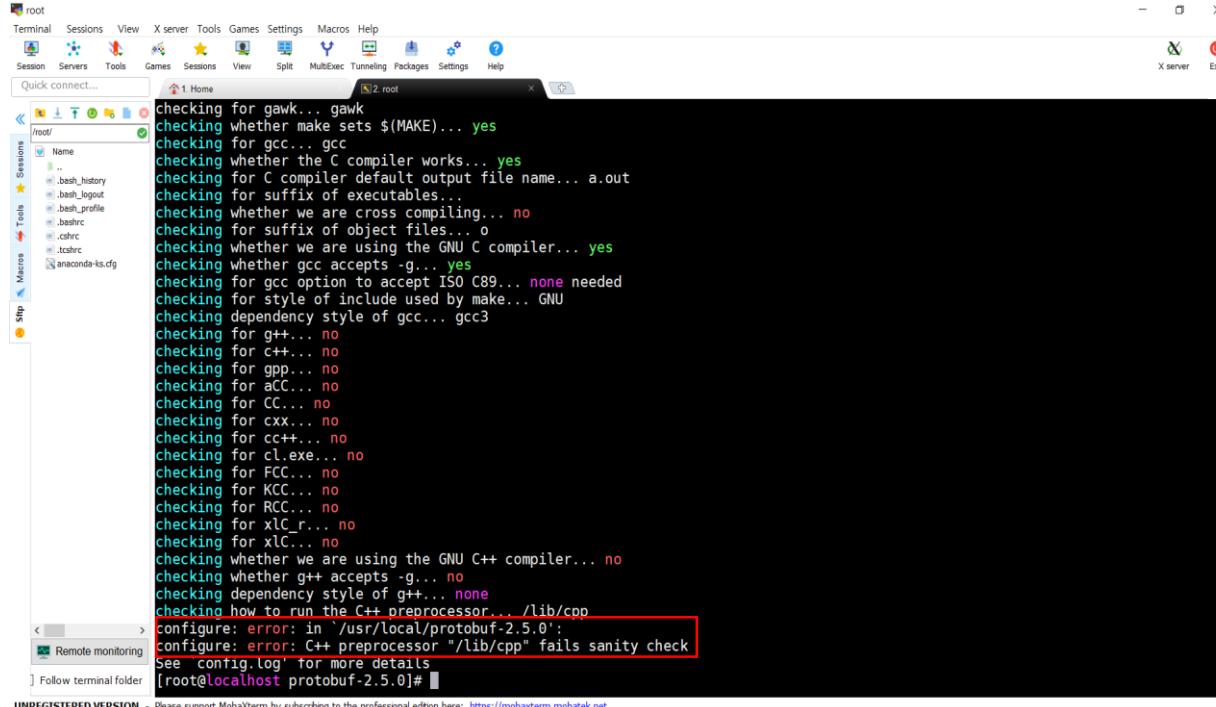
Dependency Updated:
  libgcc.x86_64 0:4.8.5-28.el7_5.1          libgomp.x86_64 0:4.8.5-28.el7_5.1

Complete!
[root@localhost protobuf-2.5.0]#
```

Protobuf-2.5.0 디렉토리로 이동 후 configure 파일 실행
오류 발생!!!!!! 해결하기 1

```
[root@localhost protobuf-2.5.0]# rpm -qa|grep gcc (컴파일러 설치 유무 확인)
[root@localhost protobuf-2.5.0]# yum -y install gcc (gcc 설치)
```

가상머신 실행



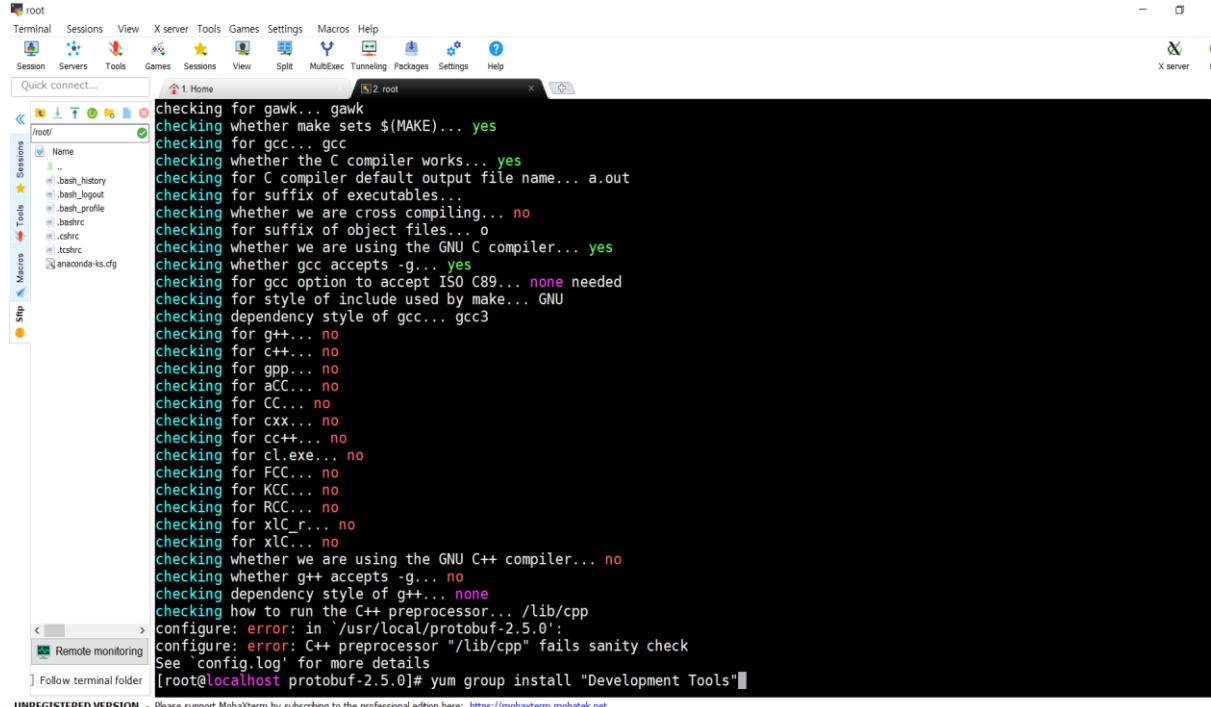
A screenshot of the MobaXterm terminal window titled 'root'. The terminal shows the execution of the 'configure' script for the protobuf-2.5.0 package. The output is as follows:

```
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
checking for g++... no
checking for c++... no
checking for gpp... no
checking for aCC... no
checking for CC... no
checking for cxx... no
checking for cc++... no
checking for cl.exe... no
checking for FCC... no
checking for KCC... no
checking for RCC... no
checking for xLC_r... no
checking for xLC... no
checking whether we are using the GNU C++ compiler... no
checking whether g++ accepts -g... no
checking dependency style of g++... none
configure: how to run the C++ preprocessor... /lib/cpp
configure: error: in `/usr/local/protobuf-2.5.0':
configure: error: C++ preprocessor "/lib/cpp" fails sanity check
See config.log for more details
[root@localhost protobuf-2.5.0]#
```

The last two lines of the output are highlighted with a red border.

gcc 설치한 후에도 configure 파일 실행 오류가 발생하면.....

가상머신 실행



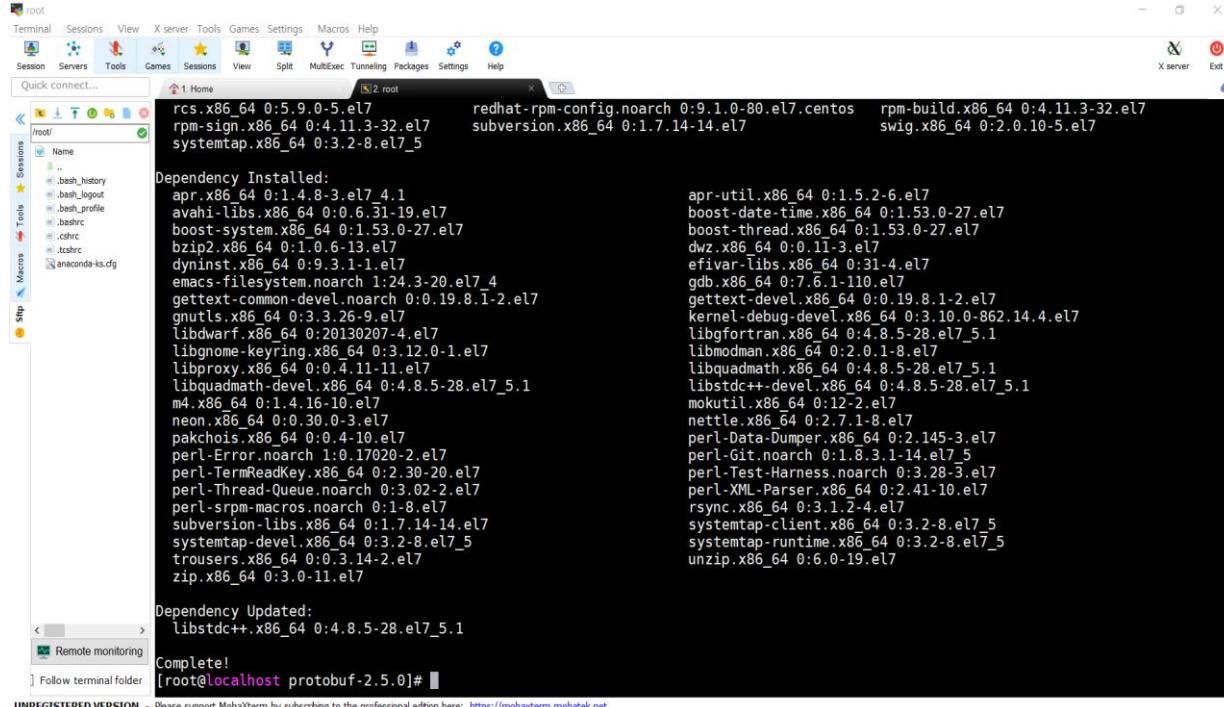
```
root@localhost ~]# ./configure
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
checking for g++... no
checking for C++... no
checking for gpp... no
checking for aCC... no
checking for CC... no
checking for cxx... no
checking for c++... no
checking for cl.exe... no
checking for FCC... no
checking for KCC... no
checking for RCC... no
checking for xLC_r... no
checking for xLC... no
checking whether we are using the GNU C++ compiler... no
checking whether g++ accepts -g... no
checking dependency style of g++... none
configure: error: in `/usr/local/protobuf-2.5.0':
configure: error: C++ preprocessor "/lib/cpp" fails sanity check
See config.log for more details
root@localhost ~]#
```

gcc 설치한 후에도 configure 파일 실행 오류가 발생하면.....

해결하기 2

```
[root@localhost protobuf-2.5.0]# yum group install "Development Tools"
```

가상머신 실행

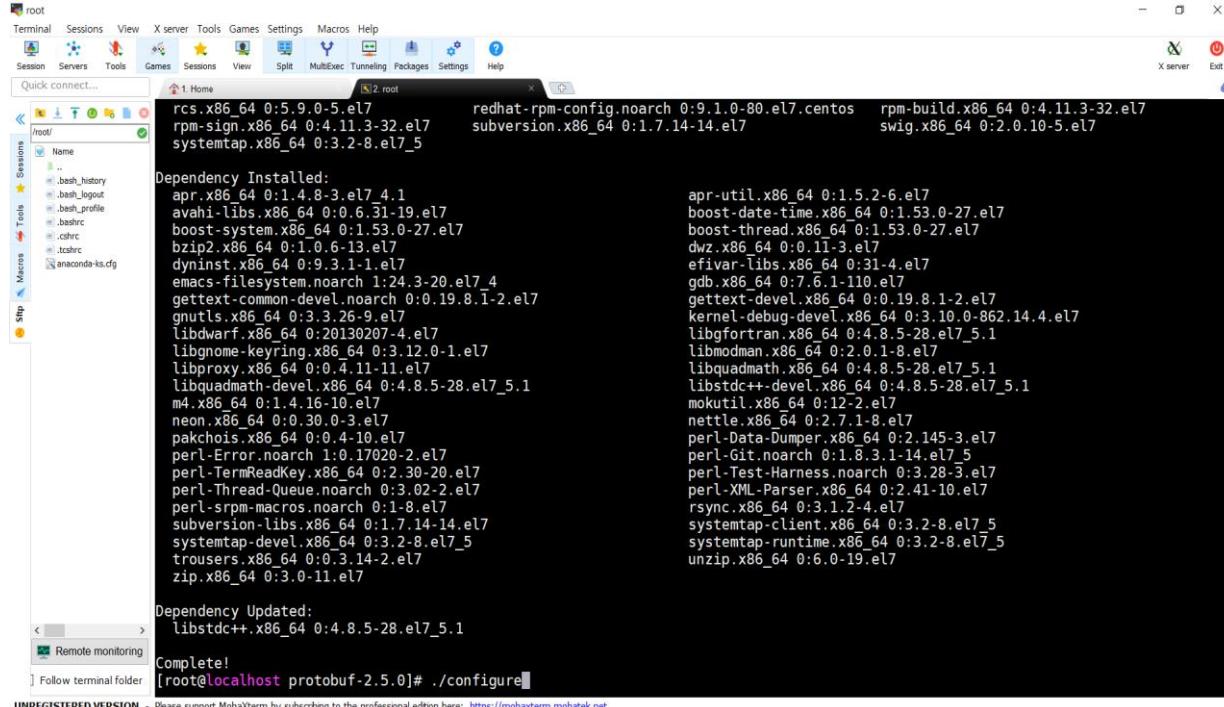


gcc 설치한 후에도 configure 파일 실행 오류가 발생하면.....

해결하기 2

```
[root@localhost protobuf-2.5.0]# yum group install "Development Tools"
```

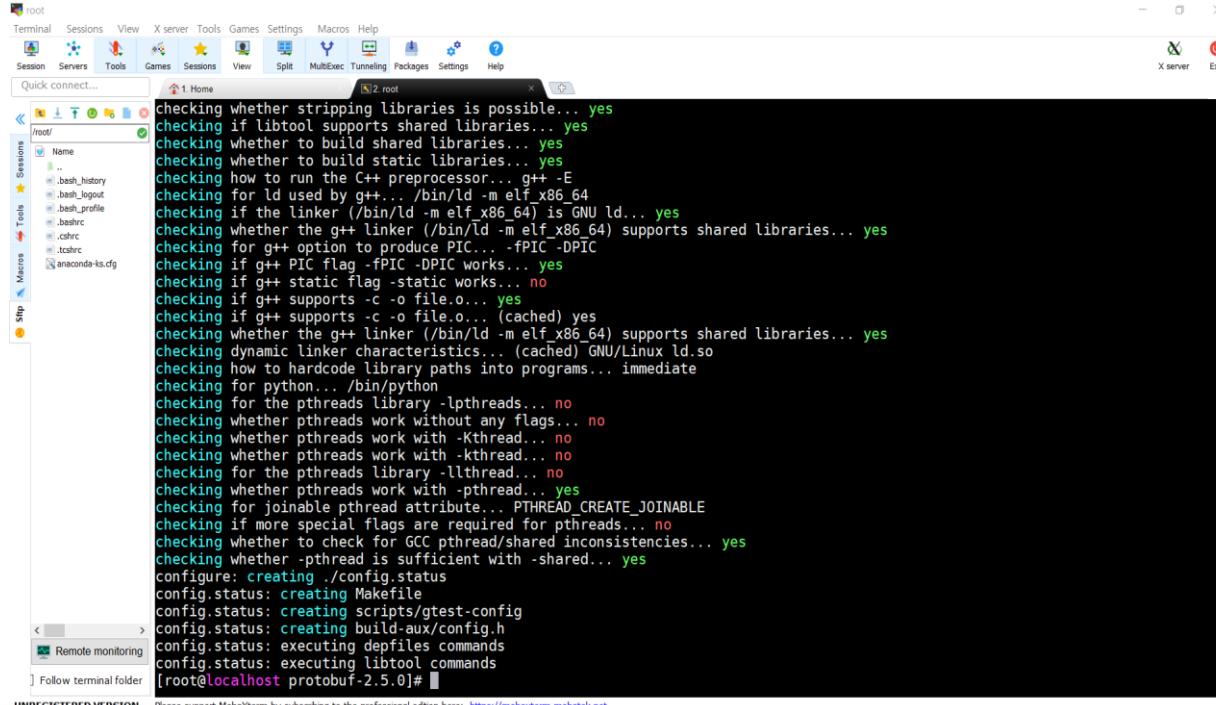
가상머신 실행



Development Tools 설치한 후 config 파일 실행

```
[root@localhost protobuf-2.5.0]# ./configure
```

가상머신 실행



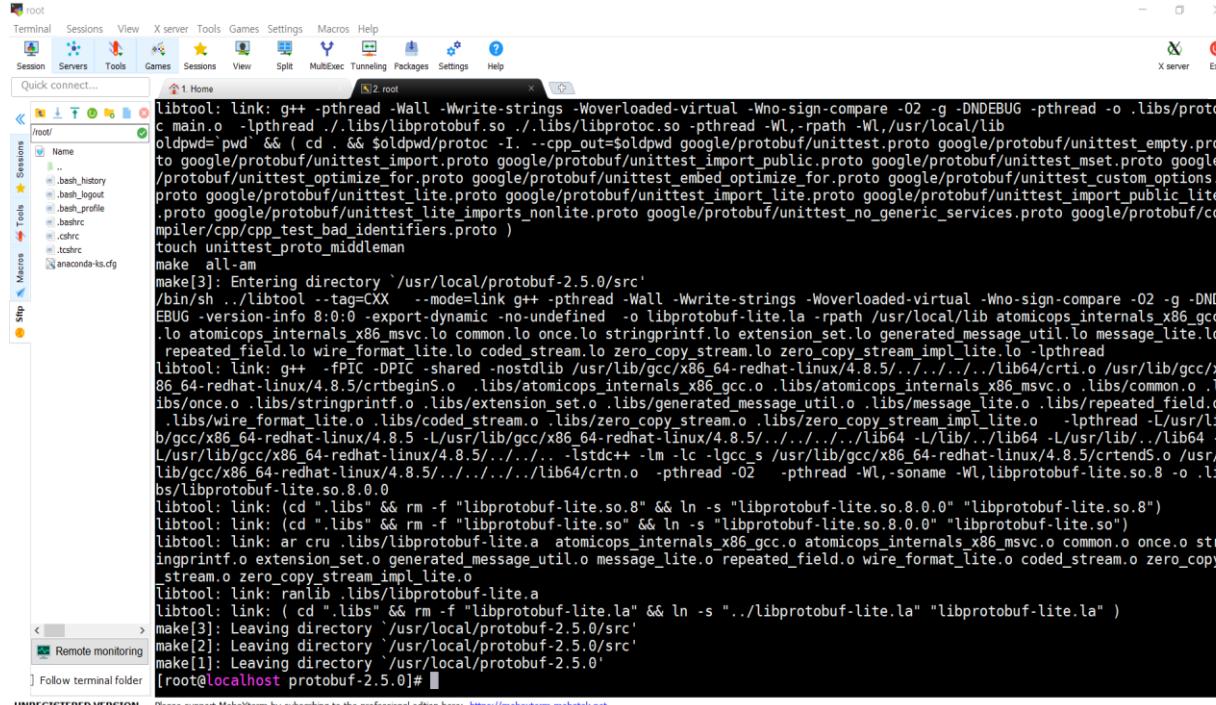
A screenshot of the MobaXterm application interface. The main window shows a terminal session titled 'root' with the command '/root/.mobaXterm/configure' being run. The output of the command is displayed in the terminal window, showing various configuration checks and their results. The terminal window has a dark background with light-colored text. The MobaXterm interface includes a sidebar with session management tools like 'Sessions', 'Tools', and 'Macros'. A status bar at the bottom indicates it's an 'UNREGISTERED VERSION'.

```
checking whether stripping libraries is possible... yes
checking if libtool supports shared libraries... yes
checking whether to build shared libraries... yes
checking whether to build static libraries... yes
checking how to run the C++ preprocessor... g++ -E
checking for ld used by g++... /bin/ld -m elf_x86_64
checking if the linker (/bin/ld -m elf_x86_64) is GNU ld... yes
checking whether the g++ linker (/bin/ld -m elf_x86_64) supports shared libraries... yes
checking for g++ option to produce PIC... -fPIC -DPIC
checking if g++ PIC flag -fPIC works... yes
checking if g++ static flag -static works... no
checking if g++ supports -c -o file.o... yes
checking if g++ supports -c -o file.o... (cached) yes
checking whether the g++ linker (/bin/ld -m elf_x86_64) supports shared libraries... yes
checking dynamic linker characteristics... (cached) GNU/Linux ld.so
checking how to hardcode library paths into programs... immediate
checking for python... /bin/python
checking for the pthreads library -lpthread... no
checking whether pthreads work without any flags... no
checking whether pthreads work with -Kthread... no
checking whether pthreads work with -kthread... no
checking for the pthreads library -llthread... no
checking whether pthreads work with -pthread... yes
checking for joinable pthread attribute... PTHREAD_CREATE_JOINABLE
checking if more special flags are required for pthreads... no
checking whether to check for GCC pthread/shared inconsistencies... yes
checking whether -pthread is sufficient with -shared... yes
configure: creating ./config.status
config.status: creating Makefile
config.status: creating scripts/gtest-config
config.status: creating build-aux/config.h
config.status: executing depfiles commands
config.status: executing libtool commands
[root@localhost protobuf-2.5.0]#
```

Development Tools 설치한 후 configure 파일 실행

```
[root@localhost protobuf-2.5.0]# ./configure
```

가상머신 실행



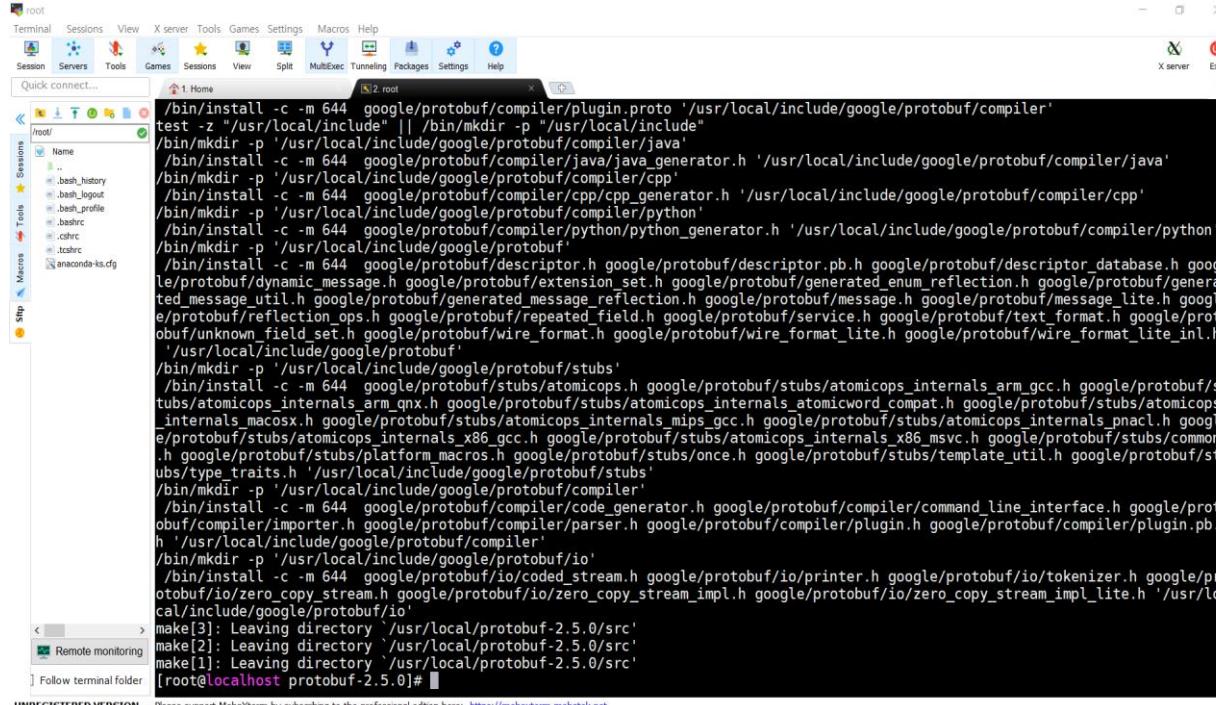
The screenshot shows a MobaXterm terminal window titled 'root' with session 'root'. The terminal displays the output of a 'make' command for libprotobuf-2.5.0. The log includes several 'libtool: link' commands, each specifying various compiler flags like '-Wall', '-Wwrite-strings', and '-DNDEBUG'. It also shows the removal of old libraries ('rm -f') and the creation of new ones ('ln -s'). The process ends with the user exiting the directory ('make[1]: Leaving directory') and exiting the terminal ('[root@localhost protobuf-2.5.0]#').

```
libtool: link: g++ -pthread -Wall -Wwrite-strings -Woverloaded-virtual -Wno-sign-compare -O2 -g -DNDEBUG -pthread -o .libs/libproto
oldpwd=`pwd` && ( cd . && $oldpwd/protoc -I . --cpp_out=$oldpwd google/protobuf/unittest.proto google/protobuf/unittest_empty.proto
to google/protobuf/unittest_import.proto google/protobuf/unittest_import_public.proto google/protobuf/unittest_mset.proto google
protobuf/unittest_optimize_for.proto google/protobuf/unittest_embed_optimize_for.proto google/protobuf/unittest_custom_options.
proto google/protobuf/unittest_lite.proto google/protobuf/unittest_import_lite.proto google/protobuf/unittest_import_public_lite
.proto google/protobuf/unittest_lite_imports_nonlite.proto google/protobuf/unittest_no_generic_services.proto google/protobuf/co
mpiler/cpp/cpp_test_bad_identifiers.proto )
touch unittest_proto_middleman
make all-am
make[3]: Entering directory `/usr/local/protobuf-2.5.0/src'
/bin/sh ..../libtool --tag=CXX --mode=link g++ -pthread -Wall -Wwrite-strings -Woverloaded-virtual -Wno-sign-compare -O2 -g -DN
DEBUG -version-info 8:0:0 -export-dynamic -no-undefined -o libprotobuf-lite.la -rpath /usr/local/lib atomicops_internals_x86_gcc
.lo atomicops_internals_x86_msvc.lo common.lo once.lo sprintf.lo extension_set.lo generated_message_util.lo message_lite.lo
repeated_field.lo wire_format_lite.lo coded_stream.lo zero_copy_stream.lo zero_copy_stream_impl_lite.lo -lpthread
libtool: link: g++ -fPIC -DPIC -shared -nostdlib /usr/lib/gcc/x86_64-redhat-linux/4.8.5/../../../../lib64/crti.o /usr/lib/gcc/x
86_64-redhat-linux/4.8.5/crtbeginS.o .libs/atomicops_internals_x86_gcc.o .libs/atomicops_internals_x86_ms
vc.o .libs/common.o .l
ibs/once.o .libs/stringprintf.o .libs/extension_set.o .libs/generated_message_util.o .libs/message_lite.o .libs/repeated_field.o
.libs/wire_format_lite.o .libs/coded_stream.o .libs/zero_copy_stream.o .libs/zero_copy_stream_impl_lite.o -lpthread -L/usr/li
b/gcc/x86_64-redhat-linux/4.8.5 -L/usr/lib/gcc/x86_64-redhat-linux/4.8.5/../../../../lib64 -L/lib64 -L/usr/lib/..../lib64 -
L/usr/lib/gcc/x86_64-redhat-linux/4.8.5/../../../../lib64/crt0.o -lstdc++ -lm -lc -lgcc_s /usr/lib/gcc/x86_64-redhat-linux/4.8.5/crtendS.o /usr/
lib/gcc/x86_64-redhat-linux/4.8.5/../../../../lib64/crtn.o -pthread -O2 -pthread -Wl,-soname -Wl,libprotobuf-lite.so.8 -o .li
bs/libprotobuf-lite.so.8.0.0
libtool: link: (cd ".libs" && rm -f "libprotobuf-lite.so.8" && ln -s "libprotobuf-lite.so.8.0.0" "libprotobuf-lite.so.8")
libtool: link: (cd ".libs" && rm -f "libprotobuf-lite.so" && ln -s "libprotobuf-lite.so.8.0.0" "libprotobuf-lite.so")
libtool: link: ar cru .libs/libprotobuf-lite.a atomicops_internals_x86_gcc.o atomicops_internals_x86_ms
vc.o common.o once.o str
ingprintf.o extension_set.o generated_message_util.o message_lite.o repeated_field.o wire_format_lite.o coded_stream.o zero_copy
_stream.o zero_copy_stream_impl_lite.o
libtool: link: ranlib .libs/libprotobuf-lite.a
libtool: link: ( cd ".libs" && rm -f "libprotobuf-lite.la" && ln -s "../libprotobuf-lite.la" "libprotobuf-lite.la" )
make[3]: Leaving directory `/usr/local/protobuf-2.5.0/src'
make[2]: Leaving directory `/usr/local/protobuf-2.5.0/src'
make[1]: Leaving directory `/usr/local/protobuf-2.5.0'
[root@localhost protobuf-2.5.0]#
```

Development Tools 설치한 후 configure 파일 실행이 정상적으로 된다면 make 명령어 실행

[root@localhost protobuf-2.5.0]# make

가상머신 실행



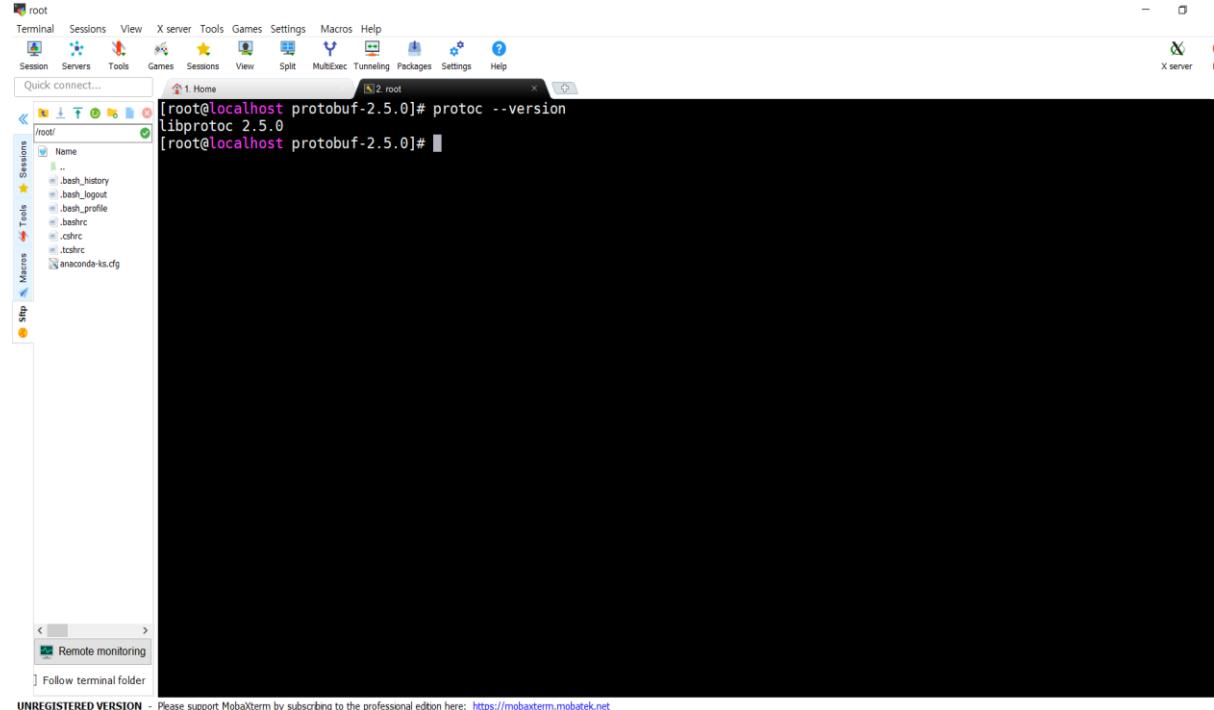
A screenshot of the MobaXterm terminal window titled 'root'. The terminal shows the output of the 'make install' command for the protobuf-2.5.0 package. The output includes numerous file creation and permission setting commands, such as '/bin/install -c -m 644 google/protobuf/compiler/plugin.proto /usr/local/include/google/protobuf/compiler', '/bin/mkdir -p "/usr/local/include"', and '/bin/install -c -m 644 google/protobuf/compiler/java_generator.h /usr/local/include/google/protobuf/compiler/java'. The terminal interface includes a sidebar with session management, tools like 'Remote monitoring', and a bottom status bar indicating it's an 'UNREGISTERED VERSION'.

```
/bin/install -c -m 644 google/protobuf/compiler/plugin.proto '/usr/local/include/google/protobuf/compiler'
test -z "/usr/local/include" || /bin/mkdir -p "/usr/local/include"
/bin/mkdir -p '/usr/local/include/google/protobuf/compiler/java'
/bin/install -c -m 644 google/protobuf/compiler/java/java_generator.h '/usr/local/include/google/protobuf/compiler/java'
/bin/mkdir -p '/usr/local/include/google/protobuf/compiler/cpp'
/bin/install -c -m 644 google/protobuf/compiler/cpp/cpp_generator.h '/usr/local/include/google/protobuf/compiler/cpp'
/bin/mkdir -p '/usr/local/include/google/protobuf/compiler/python'
/bin/install -c -m 644 google/protobuf/compiler/python/python_generator.h '/usr/local/include/google/protobuf/compiler/python'
/bin/mkdir -p '/usr/local/include/google/protobuf'
/bin/install -c -m 644 google/protobuf/descriptor.h google/protobuf/descriptor.pb.h google/protobuf/descriptor_database.h google/protobuf/dynamic_message.h google/protobuf/extension_set.h google/protobuf/generated_enum_reflection.h google/protobuf/generator_message_util.h google/protobuf/generated_message_reflection.h google/protobuf/message.h google/protobuf/message_lite.h google/protobuf/reflection_ops.h google/protobuf/repeated_field.h google/protobuf/service.h google/protobuf/text_format.h google/protobuf/unknown_field_set.h google/protobuf/wire_format.h google/protobuf/wire_format_lite.h google/protobuf/wire_format_lite_inl.h
'/usr/local/include/google/protobuf'
/bin/mkdir -p '/usr/local/include/google/protobuf/stubs'
/bin/install -c -m 644 google/protobuf/stubs/atomicops.h google/protobuf/stubs/atomicops_internals_arm_gcc.h google/protobuf/stubs/atomicops_internals_arm_qnx.h google/protobuf/stubs/atomicops_internals_atomicword_compat.h google/protobuf/stubs/atomicops_internals_macosx.h google/protobuf/stubs/atomicops_internals_mips_gcc.h google/protobuf/stubs/atomicops_internals_pnacl.h google/protobuf/stubs/atomicops_internals_x86_gcc.h google/protobuf/stubs/atomicops_internals_x86_msvc.h google/protobuf/stubs/common.h google/protobuf/stubs/platform_macros.h google/protobuf/stubs/once.h google/protobuf/stubs/template_util.h google/protobuf/stubs/type_traits.h '/usr/local/include/google/protobuf/stubs'
/bin/mkdir -p '/usr/local/include/google/protobuf/compiler'
/bin/install -c -m 644 google/protobuf/compiler/code_generator.h google/protobuf/compiler/command_line_interface.h google/protobuf/compiler/importer.h google/protobuf/compiler/parser.h google/protobuf/compiler/plugin.h google/protobuf/compiler/plugin.pb.h '/usr/local/include/google/protobuf/compiler'
/bin/mkdir -p '/usr/local/include/google/protobuf/io'
/bin/install -c -m 644 google/protobuf/io/coded_stream.h google/protobuf/io/printer.h google/protobuf/io/tokenizer.h google/protobuf/io/zero_copy_stream.h google/protobuf/io/zero_copy_stream_impl.h google/protobuf/io/zero_copy_stream_impl_lite.h '/usr/local/include/google/protobuf/io'
make[3]: Leaving directory '/usr/local/protobuf-2.5.0/src'
make[2]: Leaving directory '/usr/local/protobuf-2.5.0/src'
make[1]: Leaving directory '/usr/local/protobuf-2.5.0/src'
[root@localhost protobuf-2.5.0]#
```

Development Tools 설치한 후 configure 파일 실행이 정상적으로 된다면 make install 명령어 실행

```
[root@localhost protobuf-2.5.0]# make install
```

가상머신 실행



Protocol buffer 버전 확인

```
[root@localhost protobuf-2.5.0]# protoc --version
```

가상머신 실행

Index of /dist/hadoop/core/hadoop-2.6.0

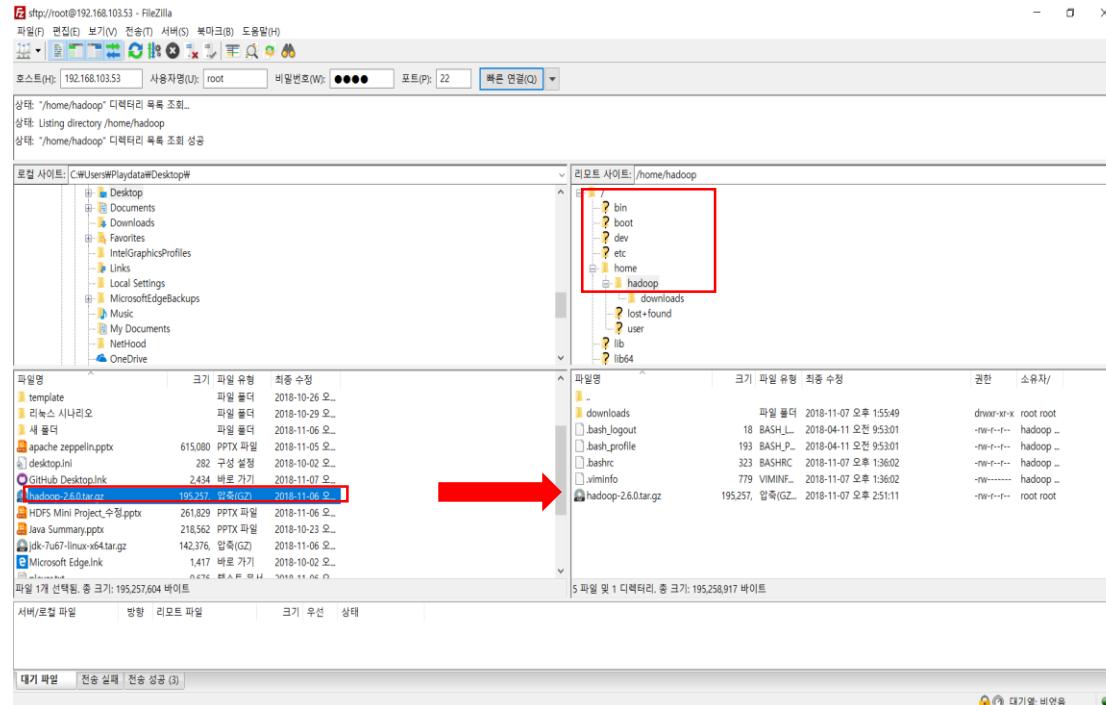
| Name | Last modified | Size | Description |
|---|------------------|------|-------------|
| Parent Directory | | - | |
| hadoop-2.6.0-src.tar.gz | 2014-11-30 23:52 | 17M | |
| hadoop-2.6.0-src.tar.gz.asc | 2014-11-30 23:52 | 833 | |
| hadoop-2.6.0-src.tar.gz.md5 | 2014-11-30 23:52 | 133 | |
| hadoop-2.6.0-src.tar.gz.mds | 2014-11-30 23:52 | 1.1K | |
| hadoop-2.6.0.tar.gz | 2014-11-30 23:52 | 186M | |
| hadoop-2.6.0.tar.gz.asc | 2014-11-30 23:52 | 833 | |
| hadoop-2.6.0.tar.gz.md5 | 2014-11-30 23:52 | 125 | |
| hadoop-2.6.0.tar.gz.mds | 2014-11-30 23:52 | 968 | |

하둡 2.6.0 버전 다운로드 받은 후
"Filezilla"를 통해 /home/hadoop에 업로드

Url
<https://archive.apache.org/dist/hadoop/core/hadoop-2.6.0/>

File
hadoop-2.6.0.tar.gz, 186MB

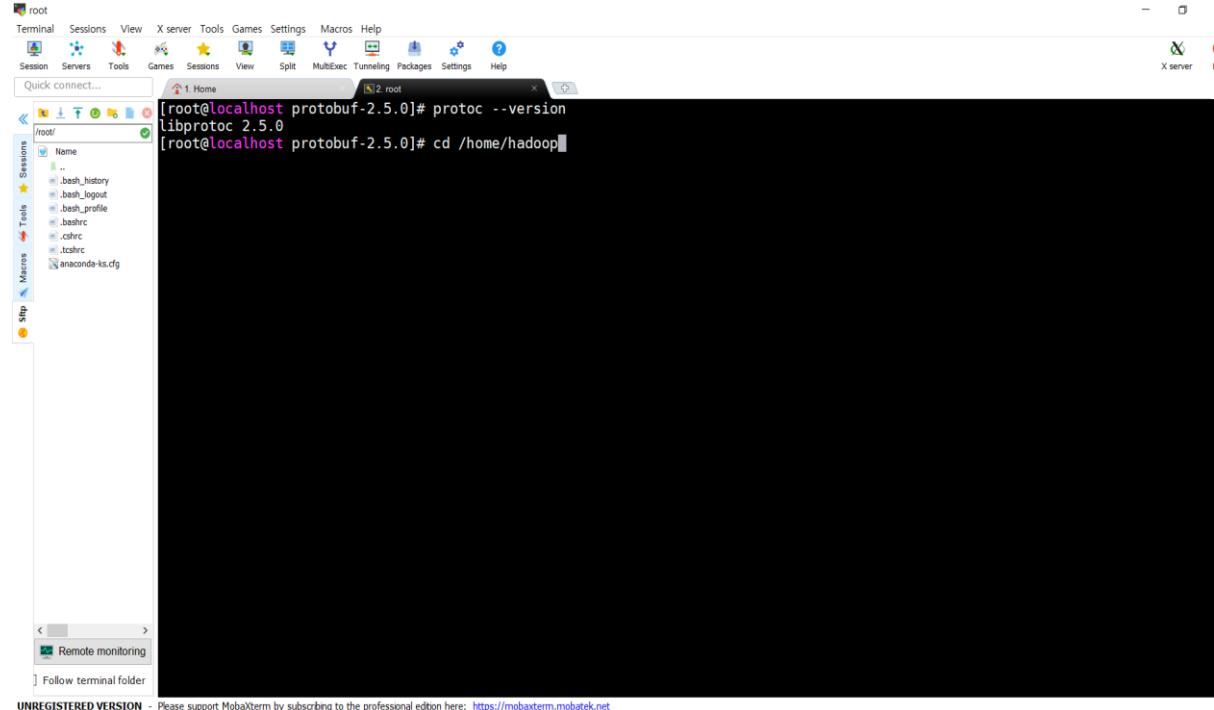
가상머신 실행



Filezilla 실행

hadoop 다운로드 사이트에서 다운로드 받은 hadoop-2.6.0.tar.gz 파일을 가상머신의 /home/hadoop 디렉토리에 복사

가상머신 실행



MobaXterm interface showing a terminal session as root on localhost. The terminal window displays the following commands:

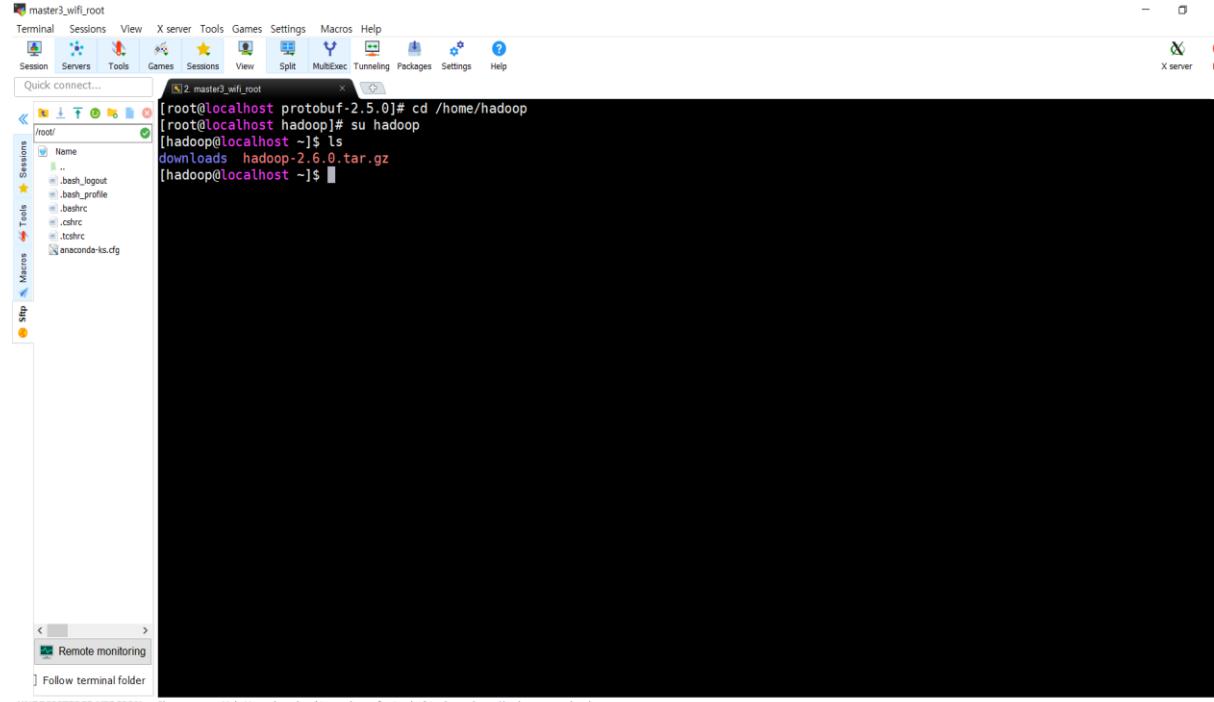
```
[root@localhost protobuf-2.5.0]# protoc --version
libprotoc 2.5.0
[root@localhost protobuf-2.5.0]# cd /home/hadoop
```

The terminal window has tabs labeled "1. Home" and "2. root". The left sidebar shows sessions and tools. The bottom status bar indicates it's an UNREGISTERED VERSION.

/home/hadoop 디렉토리로 이동

```
[root@localhost protobuf-2.5.0]# cd /home/hadoop
```

가상머신 실행



A screenshot of the MobaXterm application interface. The main window shows a terminal session titled 'master3_wifi_root'. The terminal window has a dark background and displays the following command-line session:

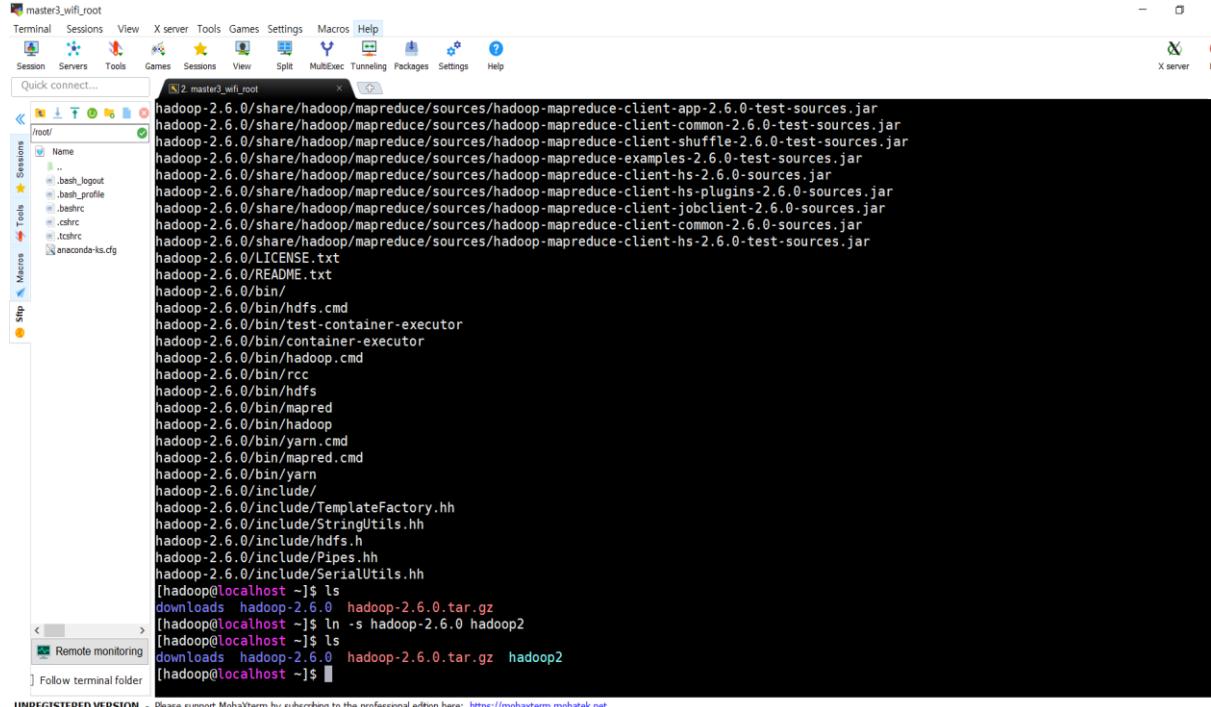
```
[root@localhost protobuf-2.5.0]# cd /home/hadoop
[root@localhost hadoop]# su hadoop
[hadoop@localhost ~]$ ls
downloads  hadoop-2.6.0.tar.gz
[hadoop@localhost ~]$
```

The MobaXterm interface includes a menu bar with options like Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, and Help. On the left, there's a sidebar with Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, and Help buttons. Below the terminal window, there are sections for 'Sessions' (with a list of saved sessions), 'Tools' (with a list of installed tools), and 'Macros'. At the bottom, there are buttons for 'Remote monitoring' and 'Follow terminal folder'. A status bar at the very bottom indicates 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.

root 계정에서 hadoop 계정으로 계정 사용 변경

```
[root@localhost hadoop]# su hadoop
[hadoop@localhost ~]$ ls
```

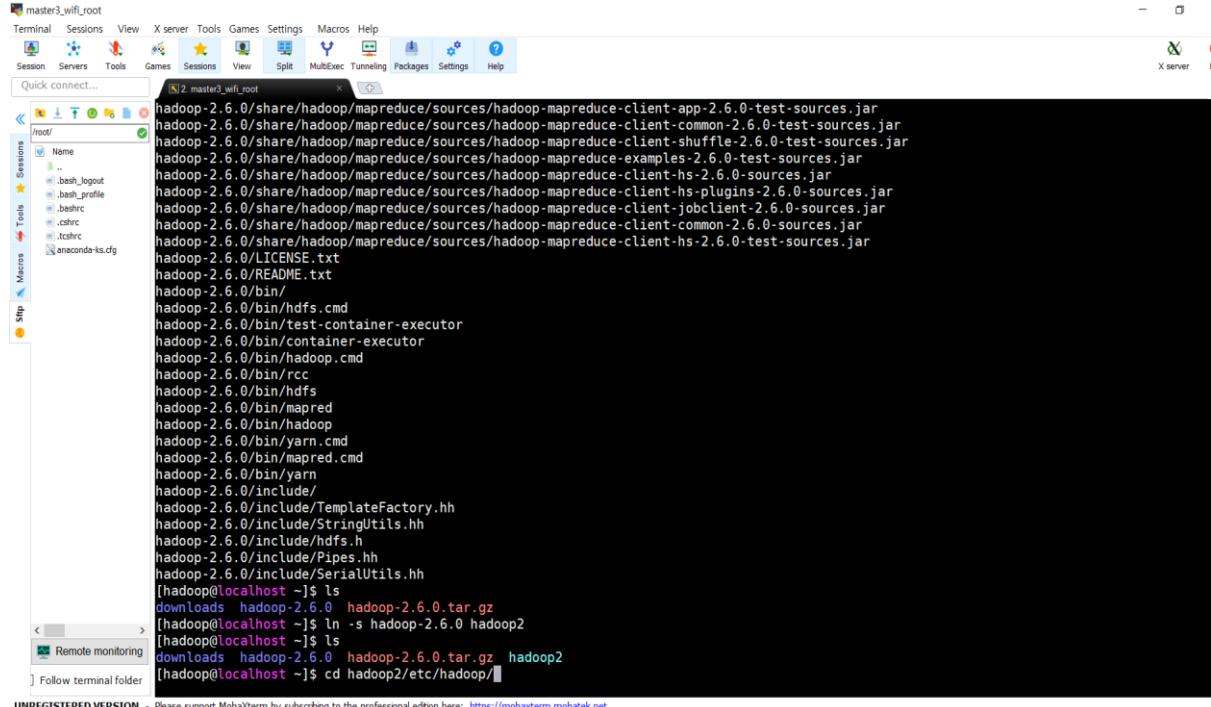
가상머신 실행



hadoop-2.6.0.tar.gz 파일 압축 해제 및 명칭 변경

```
[hadoop@localhost ~]$ tar xvzf hadoop-2.6.0.tar.gz (파일 압축 해제)  
[hadoop@localhost ~]$ ln -s hadoop-2.6.0 hadoop2 (명칭 변경)
```

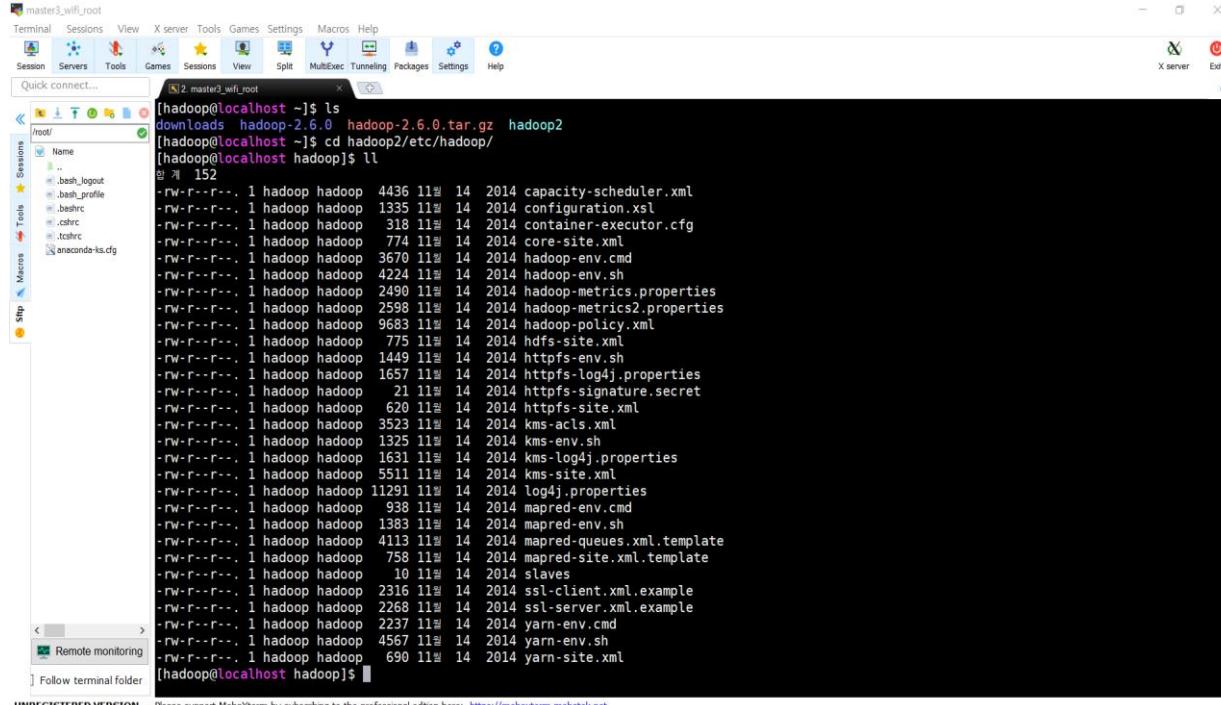
가상머신 실행



hadoop2/etc/hadoop/ 디렉토리로 이동

```
[hadoop@localhost ~]$ cd hadoop2/etc/hadoop/
```

가상머신 실행



The screenshot shows a MobaXterm terminal window titled "master3_wifi_root". It displays the following command sequence:

```
[hadoop@localhost ~]$ ls
[hadoop@localhost ~]$ cd hadoop2/etc/hadoop/
[hadoop@localhost hadoop]$ ll
```

The "ll" command output lists numerous files in the directory, including configuration files like "core-site.xml", "mapred-site.xml", and "yarn-site.xml", as well as environment scripts like "hadoop-env.sh" and "yarn-env.sh".

hadoop2/etc/hadoop/ 디렉토리에서 "ll" 명령어로 파일 확인하고 환경설정해야 할 파일 확인

[hadoop@localhost hadoop]\$ ll

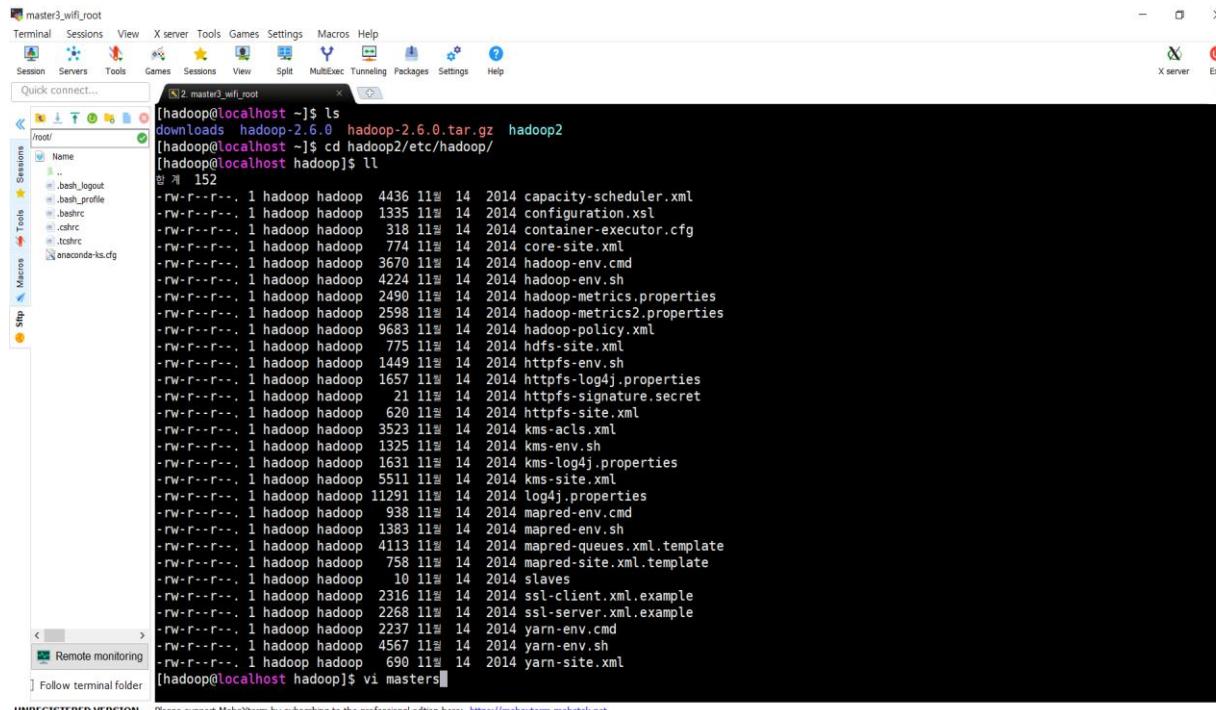
환경 설정해야 할 파일들

core-site.xml
hadoop-env.sh
hadoop-env.sh
hdfs-site.xml
mapred-site.xml.template
slaves
yarn-env.sh



가상 분산 모드로 진행
(하나의 머신에 Master와 Slave 설치)

가상 분산 모드



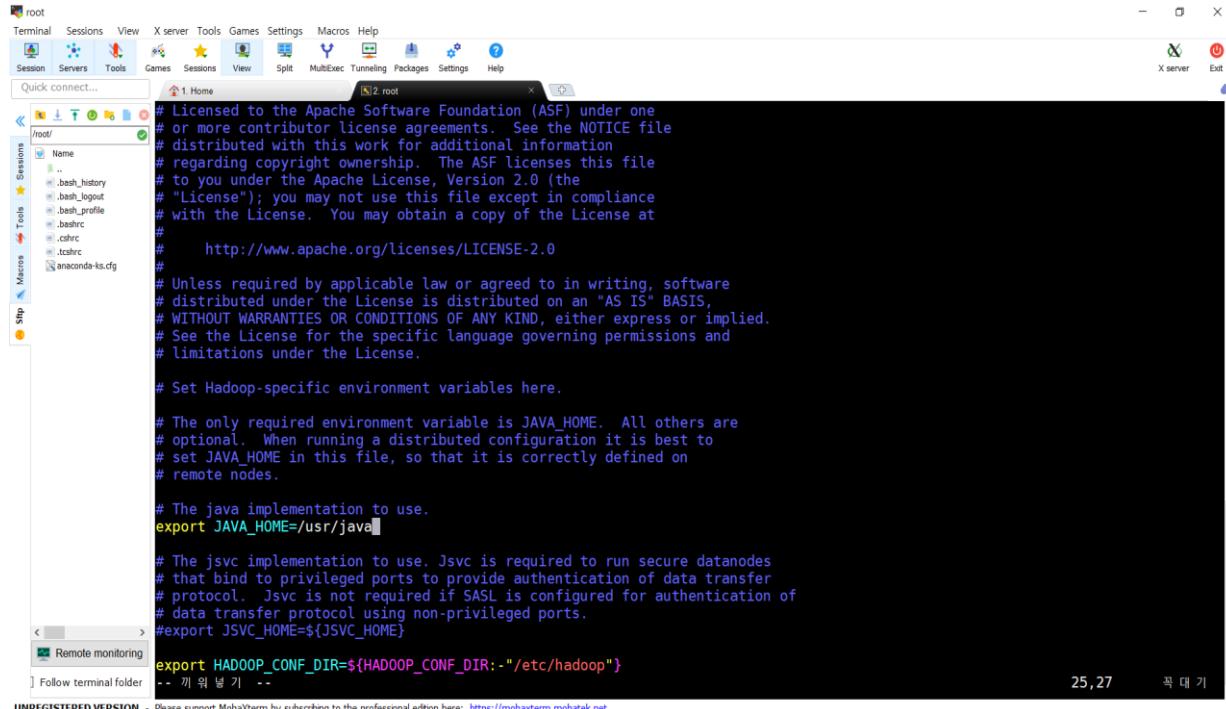
```
[hadoop@localhost ~]$ ls
downloads hadoop-2.6.0 hadoop-2.6.0.tar.gz hadoop2
[hadoop@localhost ~]$ cd hadoop2/etc/hadoop/
[hadoop@localhost hadoop]$ ll
합계 152
-rw-r--r--. 1 hadoop hadoop 4436 11월 14 2014 capacity-scheduler.xml
-rw-r--r--. 1 hadoop hadoop 1335 11월 14 2014 configuration.xml
-rw-r--r--. 1 hadoop hadoop 318 11월 14 2014 container-executor.cfg
-rw-r--r--. 1 hadoop hadoop 774 11월 14 2014 core-site.xml
-rw-r--r--. 1 hadoop hadoop 3670 11월 14 2014 hadoop-env.cmd
-rw-r--r--. 1 hadoop hadoop 4224 11월 14 2014 hadoop-env.sh
-rw-r--r--. 1 hadoop hadoop 2490 11월 14 2014 hadoop-metrics.properties
-rw-r--r--. 1 hadoop hadoop 2598 11월 14 2014 hadoop-metrics2.properties
-rw-r--r--. 1 hadoop hadoop 9683 11월 14 2014 hadoop-policy.xml
-rw-r--r--. 1 hadoop hadoop 775 11월 14 2014 hdfs-site.xml
-rw-r--r--. 1 hadoop hadoop 1449 11월 14 2014 httpsfs-env.sh
-rw-r--r--. 1 hadoop hadoop 1657 11월 14 2014 httpfs-log4j.properties
-rw-r--r--. 1 hadoop hadoop 21 11월 14 2014 httpfs-signature.secret
-rw-r--r--. 1 hadoop hadoop 620 11월 14 2014 httpfs-site.xml
-rw-r--r--. 1 hadoop hadoop 3523 11월 14 2014 kms-acls.xml
-rw-r--r--. 1 hadoop hadoop 1325 11월 14 2014 kms-env.sh
-rw-r--r--. 1 hadoop hadoop 1631 11월 14 2014 kms-log4j.properties
-rw-r--r--. 1 hadoop hadoop 5511 11월 14 2014 kms-site.xml
-rw-r--r--. 1 hadoop hadoop 11291 11월 14 2014 log4j.properties
-rw-r--r--. 1 hadoop hadoop 938 11월 14 2014 mapred-env.cmd
-rw-r--r--. 1 hadoop hadoop 1383 11월 14 2014 mapred-env.sh
-rw-r--r--. 1 hadoop hadoop 4113 11월 14 2014 mapred-queues.xml.template
-rw-r--r--. 1 hadoop hadoop 758 11월 14 2014 mapred-site.xml.template
-rw-r--r--. 1 hadoop hadoop 10 11월 14 2014 slaves
-rw-r--r--. 1 hadoop hadoop 2316 11월 14 2014 ssl-client.xml.example
-rw-r--r--. 1 hadoop hadoop 2268 11월 14 2014 ssl-server.xml.example
-rw-r--r--. 1 hadoop hadoop 2237 11월 14 2014 yarn-env.cmd
-rw-r--r--. 1 hadoop hadoop 4567 11월 14 2014 yarn-env.sh
-rw-r--r--. 1 hadoop hadoop 690 11월 14 2014 yarn-site.xml
[hadoop@localhost hadoop]$ vi masters
```

hadoop2/etc/hadoop/ 디렉토리에서 vi 편집기로
masters 파일 생성
파일 내 내용은 현재 사용하는 host의 고정 ip 입력

```
[hadoop@localhost hadoop]$ vi masters
```



가상 분산 모드



```
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements. See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership. The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License. You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Set Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME. All others are
# optional. When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=/usr/java

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol. Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}
```

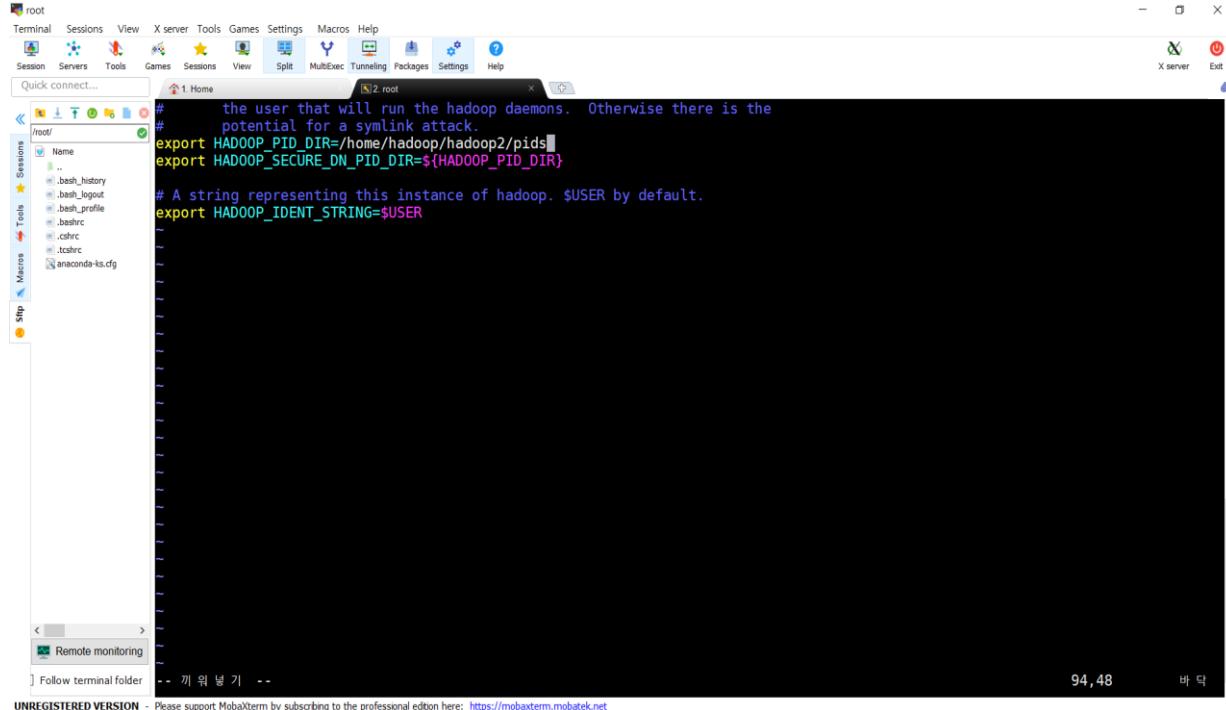
hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로
hadoop-env.sh 파일 편집

[hadoop@localhost hadoop]\$ vim hadoop-env.sh

"JAVA_HOME=\${JAVA_HOME}"을 "JAVA_HOME=/usr/java"로 변경

"HADOOP_PID_DIR=\${HADOOP_PID_DIR}"을
"HADOOP_PID_DIR=/home/hadoop/hadoop2/pids"로 변경

가상 분산 모드



```
# the user that will run the hadoop daemons. Otherwise there is the
# potential for a symlink attack.
export HADOOP_PID_DIR=/home/hadoop/hadoop2/pids
export HADOOP_SECURE_DN_PID_DIR=${HADOOP_PID_DIR}

# A string representing this instance of hadoop. $USER by default.
export HADOOP_IDENT_STRING=$USER
```

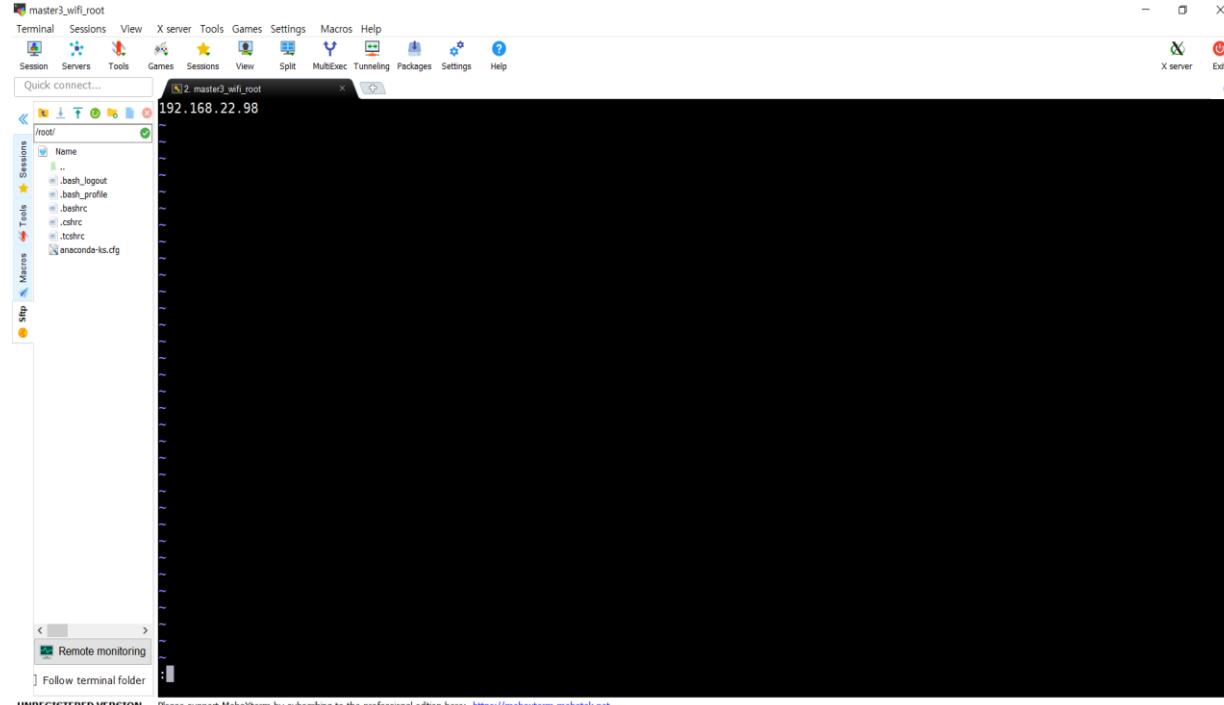
hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로
hadoop-env.sh 파일 편집

[hadoop@localhost hadoop]\$ vim hadoop-env.sh

"JAVA_HOME=\${JAVA_HOME}"을 "JAVA_HOME=/usr/java"로 변경

"HADOOP_PID_DIR=\${HADOOP_PID_DIR}"을
"HADOOP_PID_DIR=/home/hadoop/hadoop2/pids"로 변경

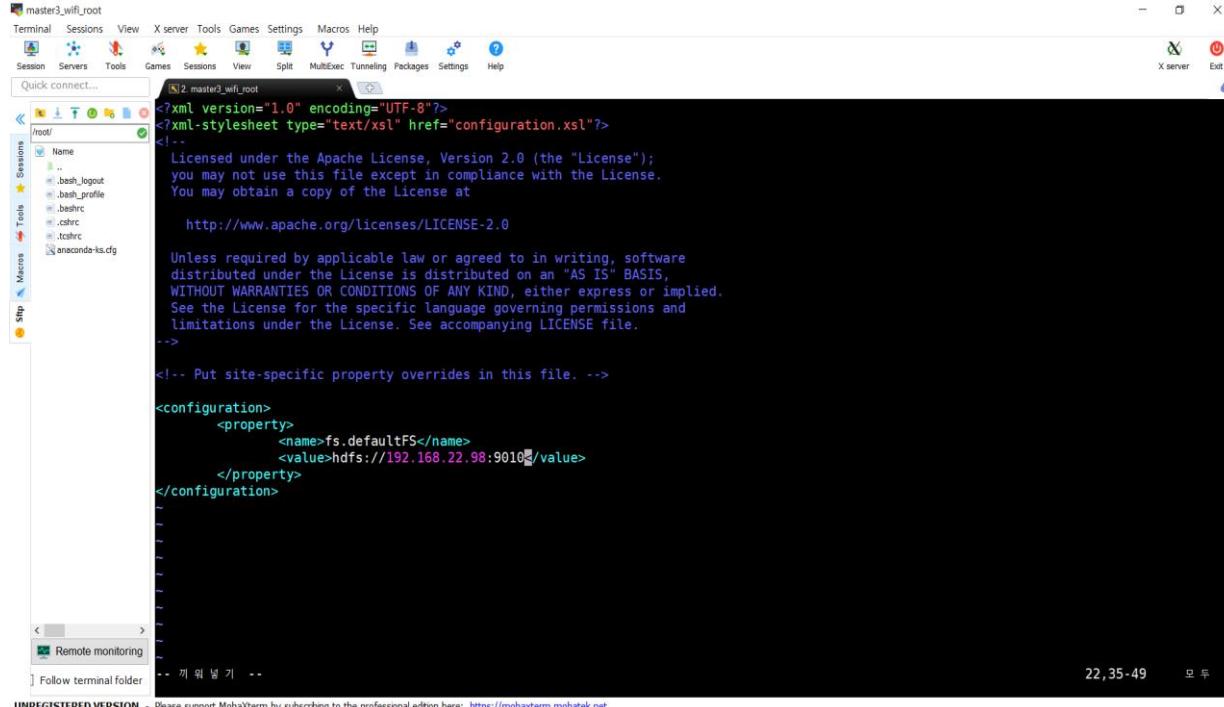
가상 분산 모드



hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로
slaves 파일 편집
파일 내 내용은 현재 사용하는 host의 고정 ip 입력

```
[hadoop@localhost hadoop]$ vim slaves
```

가상 분산 모드



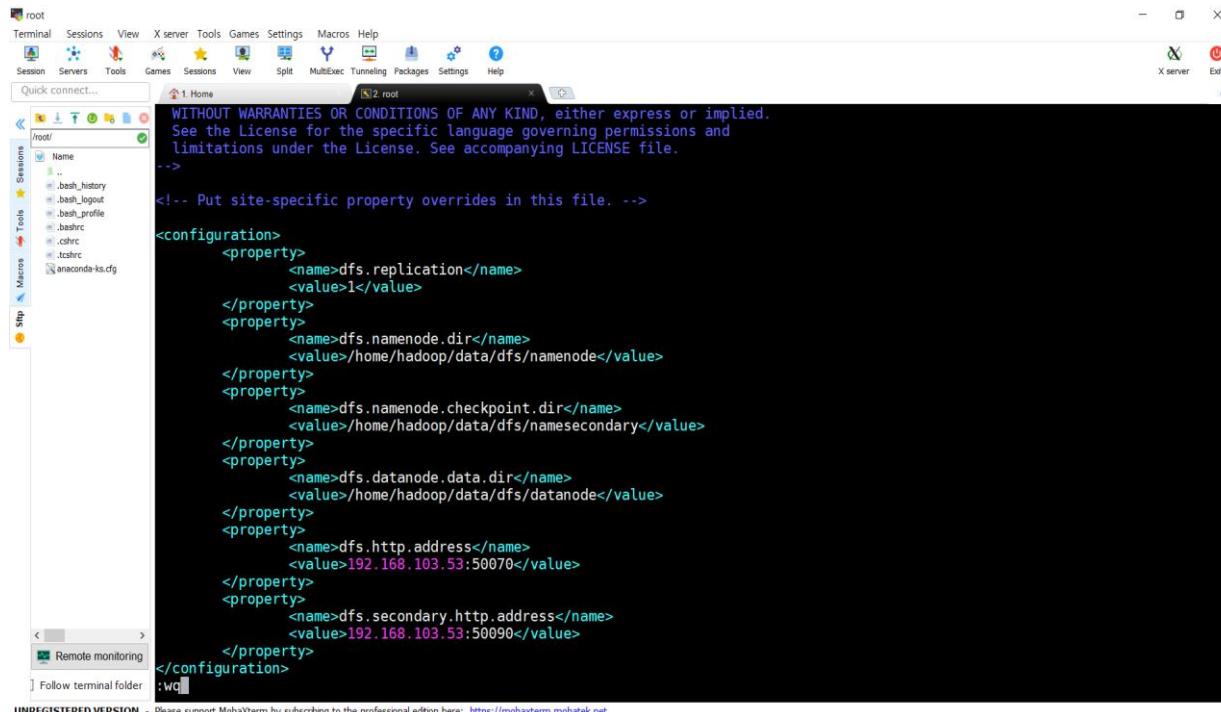
hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로 core-site.xml 파일 편집

```
[hadoop@localhost hadoop]$ vim core-site.xml
```

* <configuration></configuration> 부분에서 안 부분을 수정해야함

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://호스트의 아이피:9010</value>
    </property>
</configuration>
```

가상 분산 모드



hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로 hdfs-site.xml 파일 편집

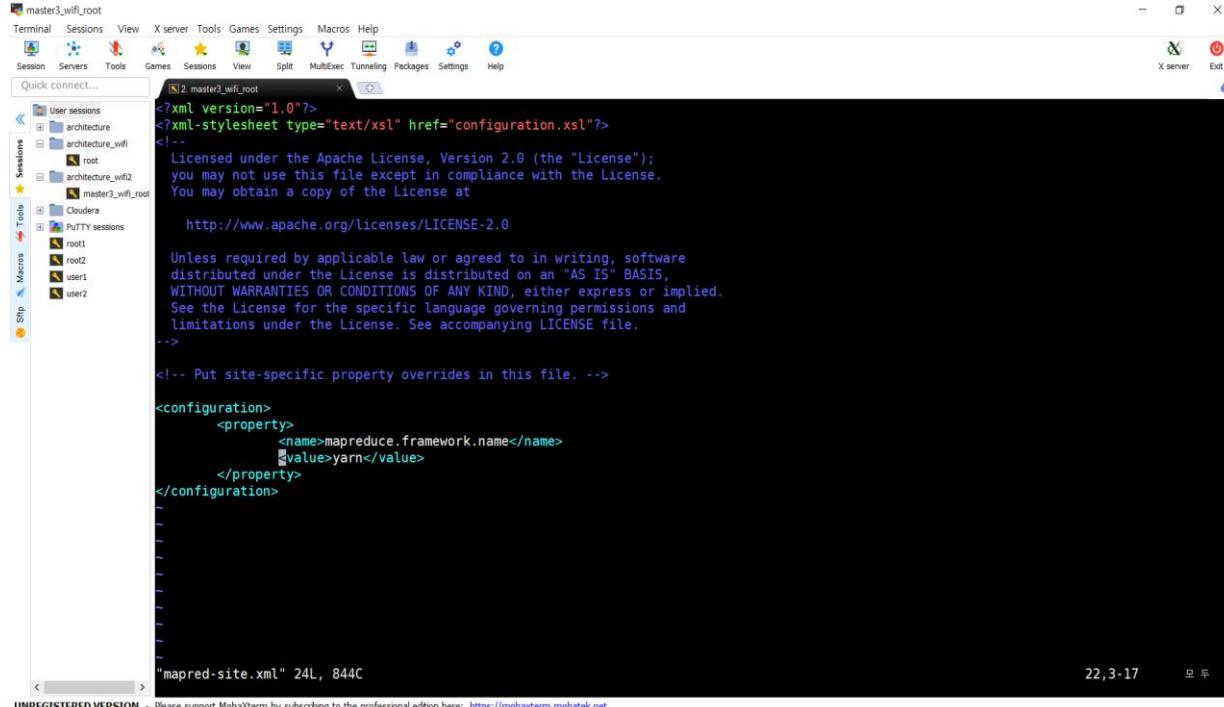
```
[hadoop@localhost hadoop]$ vim hdfs-site.xml
```

* <configuration></configuration> 부분에서 안 부분을 수정해야함

다음과 같음

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1 </value> (1 : 복제본이 하나라는 뜻 -> 수행과제는 2로 해야함)
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/home/hadoop/data/dfs/namenode</value>
  </property>
  <property>
    <name>dfs.namenode.checkpoint.dir</name>
    <value>/home/hadoop/data/dfs/namesecondary</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/home/hadoop/data/dfs/datanode</value>
  </property>
  <property>
    <name>dfs.http.address</name>
    <value>고정 ip:50070</value>
  </property>
  <property>
    <name>dfs.secondary.http.address</name>
    <value>고정 ip:50090</value>
  </property>
</configuration>
```

가상 분산 모드



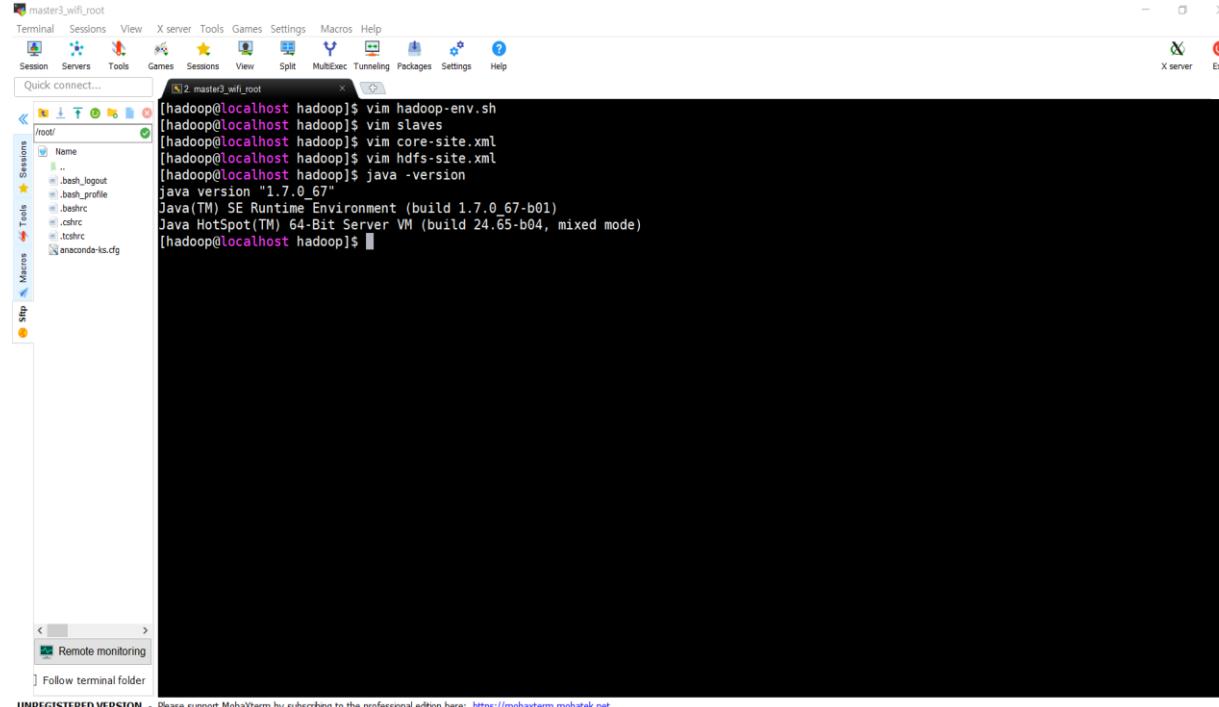
hadoop2/etc/hadoop/ 디렉토리에서 mapred-site.xml.template를 mapred-site.xml로 mv 이후 vim 편집기로 mapred-site.xml 파일 편집

```
[hadoop@localhost hadoop]$ mv mapred-site.xml.template mapred-site.xml  
[hadoop@localhost hadoop]$ vim mapred-site.xml
```

* <configuration></configuration> 부분에서 안 부분을 수정해야함

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
</configuration>
```

가상 분산 모드

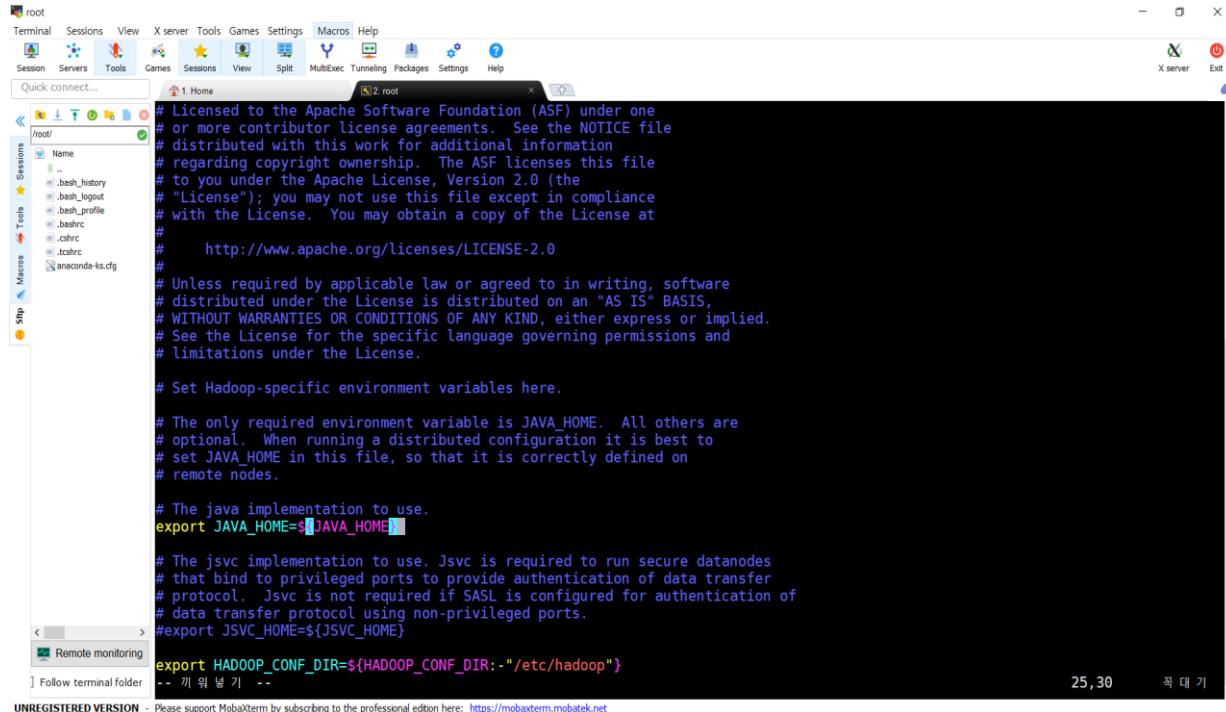


```
[hadoop@localhost hadoop]$ vim hadoop-env.sh
[hadoop@localhost hadoop]$ vim slaves
[hadoop@localhost hadoop]$ vim core-site.xml
[hadoop@localhost hadoop]$ vim hdfs-site.xml
[hadoop@localhost hadoop]$ java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
[hadoop@localhost hadoop]$
```

[hadoop@localhost hadoop]\$ java -version
java 버전 확인 발생

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

가상 분산 모드



```
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements. See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership. The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License. You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Set Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME. All others are
# optional. When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=${JAVA_HOME}

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol. Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}
```

hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로
hadoop-env.sh 파일 편집

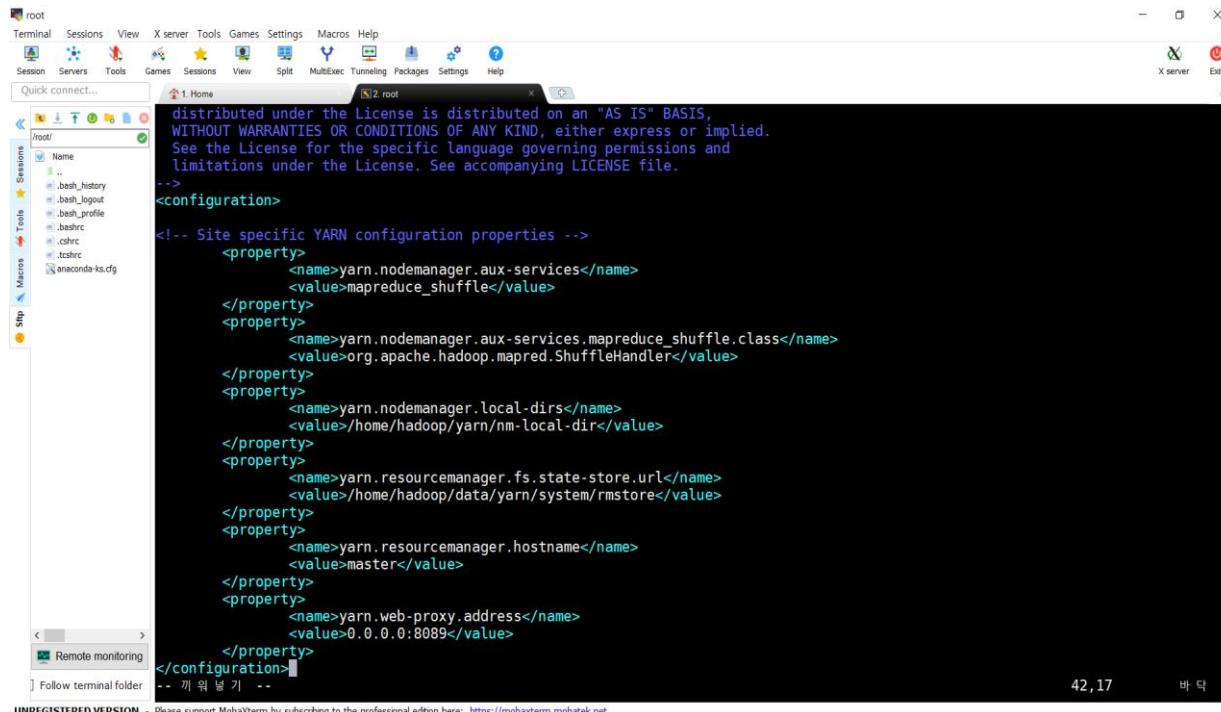
[hadoop@localhost hadoop]\$ vim hadoop-env.sh

JAVA_HOME 부분 수정

export JAVA_HOME=/usr/java를

export JAVA_HOME=\${JAVA_HOME}으로 변경

가상 분산 모드



hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로 yarn-site.xml 파일 편집

[hadoop@localhost hadoop]\$ vim yarn-site.xml

<configuration></configuration> 부분 안에 수정해야함

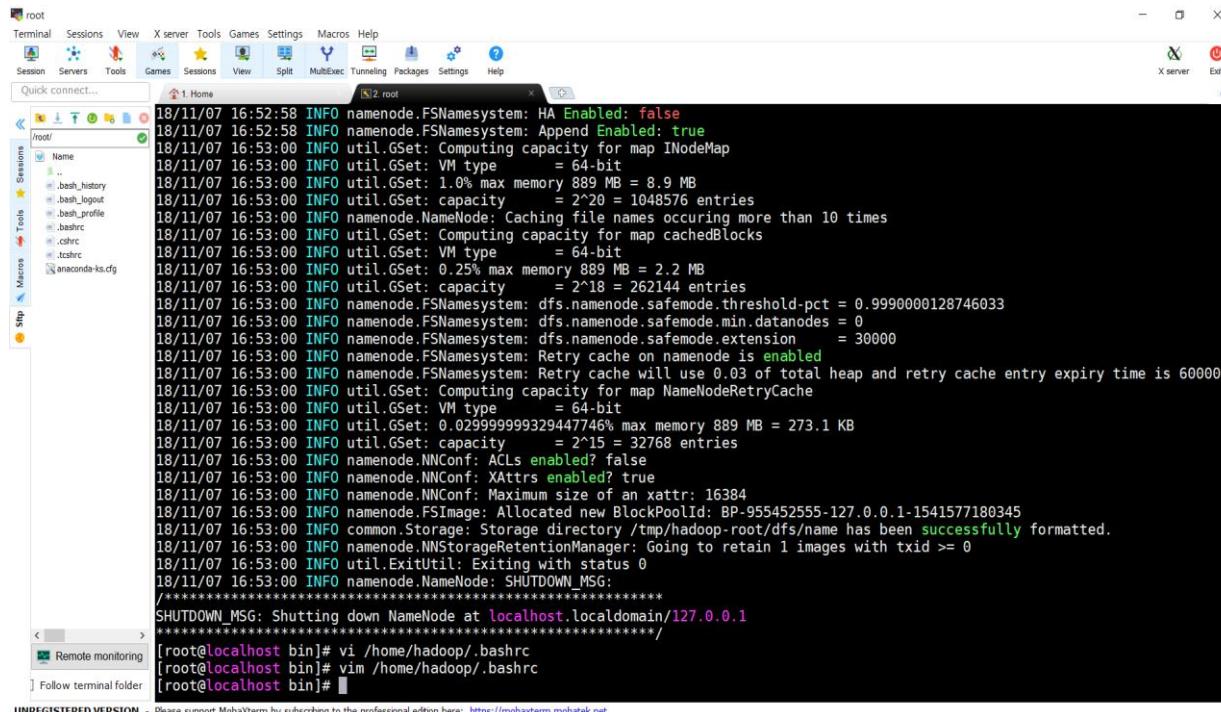
```
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.nodemanager.local-dirs</name>
  <value>/home/hadoop/yarn/nm-local-dir</value>
</property>
<property>
  <name>yarn.resourcemanager.fs.state-store.url</name>
  <value>/home/hadoop/data/yarn/system/rmstore</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>master</value>
</property>
<property>
  <name>yarn.web-proxy.address</name>
  <value>0.0.0.0:8089</value>
</property>
</configuration>
```

-- 끄 위 넣기 --

42, 17 바 닉

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

가상 분산 모드



The screenshot shows a MobaXterm terminal window titled 'root' with session '1. Home'. The terminal displays log output from the HDFS namenode format command. Key logs include:

```
18/11/07 16:52:58 INFO namenode.FSNamesystem: HA Enabled: false
18/11/07 16:52:58 INFO namenode.FSNamesystem: Append Enabled: true
18/11/07 16:53:00 INFO util.GSet: Computing capacity for map INodeMap
18/11/07 16:53:00 INFO util.GSet: VM type = 64-bit
18/11/07 16:53:00 INFO util.GSet: 1.0% max memory 889 MB = 8.9 MB
18/11/07 16:53:00 INFO util.GSet: capacity = 2^20 = 1048576 entries
18/11/07 16:53:00 INFO namenode.NameNode: Caching file names occurring more than 10 times
18/11/07 16:53:00 INFO util.GSet: Computing capacity for map cachedBlocks
18/11/07 16:53:00 INFO util.GSet: VM type = 64-bit
18/11/07 16:53:00 INFO util.GSet: 0.25% max memory 889 MB = 2.2 MB
18/11/07 16:53:00 INFO util.GSet: capacity = 2^18 = 262144 entries
18/11/07 16:53:00 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
18/11/07 16:53:00 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
18/11/07 16:53:00 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension = 30000
18/11/07 16:53:00 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
18/11/07 16:53:00 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 60000
18/11/07 16:53:00 INFO util.GSet: Computing capacity for map NameNodeRetryCache
18/11/07 16:53:00 INFO util.GSet: VM type = 64-bit
18/11/07 16:53:00 INFO util.GSet: 0.02999999329447746% max memory 889 MB = 273.1 KB
18/11/07 16:53:00 INFO util.GSet: capacity = 2^15 = 32768 entries
18/11/07 16:53:00 INFO namenode.NNConf: ACLs enabled? false
18/11/07 16:53:00 INFO namenode.NNConf: XAttrs enabled? true
18/11/07 16:53:00 INFO namenode.NNConf: Maximum size of an xattr: 16384
18/11/07 16:53:00 INFO namenode.FSImage: Allocated new BlockPoolId: BP-955452555-127.0.0.1-1541577180345
18/11/07 16:53:00 INFO common.Storage: Storage directory /tmp/hadoop-root/dfs/name has been successfully formatted.
18/11/07 16:53:00 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
18/11/07 16:53:00 INFO util.ExitUtil: Exiting with status 0
18/11/07 16:53:00 INFO namenode.NameNode: SHUTDOWN_MSG:
*****{*}
SHUTDOWN_MSG: Shutting down NameNode at localhost.localdomain/127.0.0.1
[root@localhost bin]# vi /home/hadoop/.bashrc
[root@localhost bin]# vim /home/hadoop/.bashrc
[root@localhost bin]#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

hadoop2/bin 디렉토리로 이동

/home/hadoop/.bashrc 파일을 source로 실행해 적용
hdfs namenode –format 실행

```
[root@localhost hadoop]# cd hadoop2/bin
```

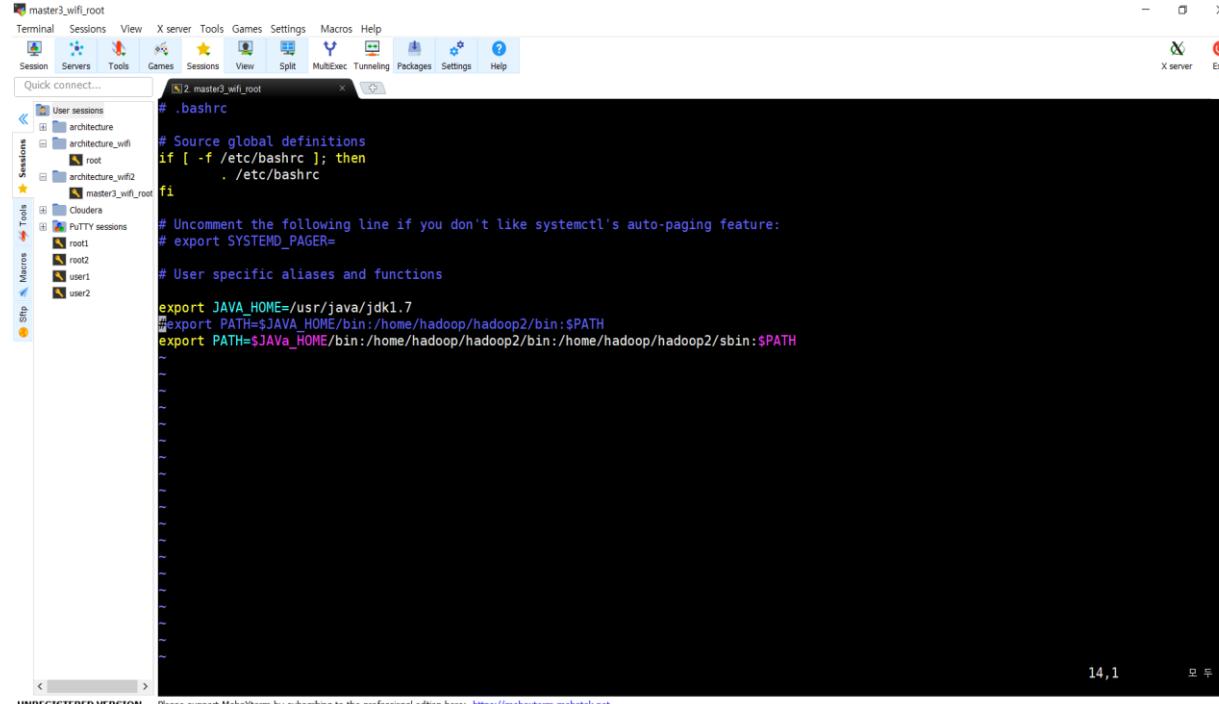
vim으로 /home/hadoop/.bashrc 에 들어가서
export PATH=\$JAVA_HOME/bin:/home/hadoop/hadoop/bin:\$PATH 를
export PATH=\$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:\$PATH 로 변경한 후

Source /home/hadoop/.bashrc 실행

```
[root@localhost bin]# source /home/hadoop/.bashrc
```

```
[root@localhost bin]# hdfs namenode -format
```

가상 분산 모드



The screenshot shows a MobaXterm window with a terminal session titled "master3_wifi_root". The session list on the left includes "User sessions" (architecture_wifi, architecture_wifi2, master3_wifi_root), "PUTTY sessions" (root1, root2, user1, user2), and "Cloudera". The terminal window displays the contents of the .bashrc file:

```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions

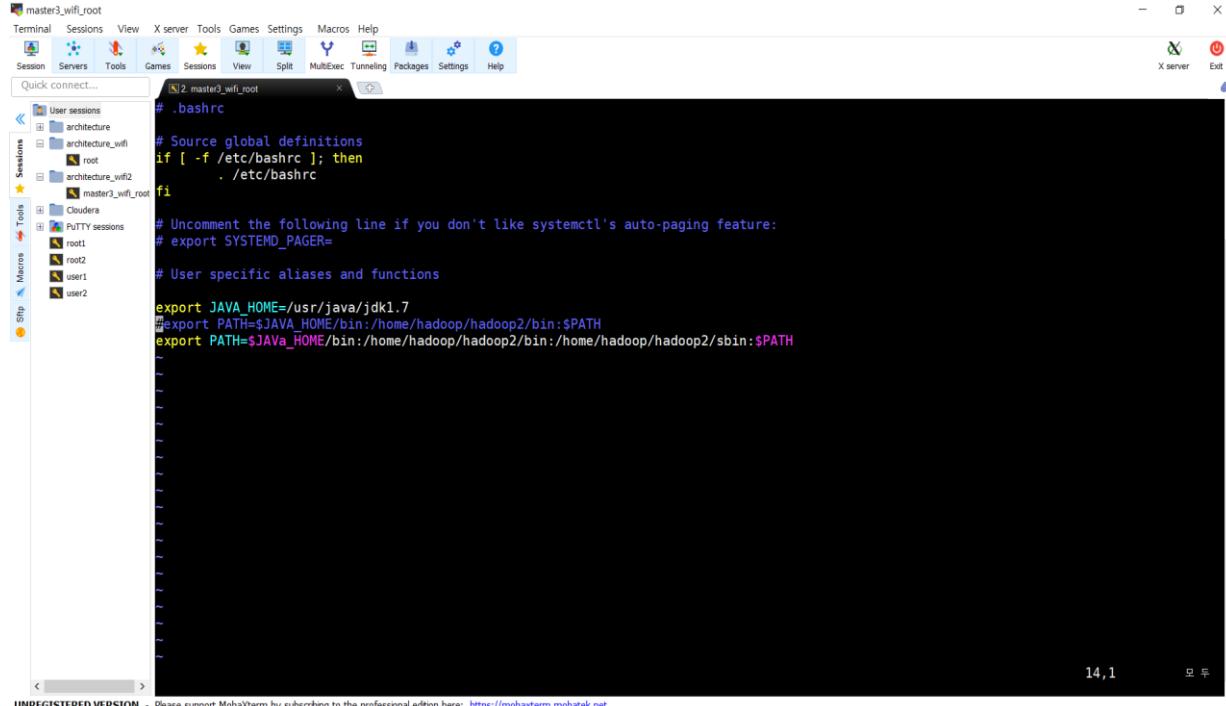
export JAVA_HOME=/usr/java/jdk1.7
#export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:$PATH
export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:/home/hadoop/hadoop2/sbin:$PATH
```

The terminal window has a dark background and light-colored text. The status bar at the bottom right shows "14,1" and "모두".

hadoop2/bin 디렉토리로에서 hadoop2/sbin 디렉토리
로 이동

```
[hadoop@localhost bin]$ cd ..
[hadoop@localhost hadoop2]$ cd sbin/
[hadoop@localhost sbin]$
```

가상 분산 모드



```
# .bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions

export JAVA_HOME=/usr/java/jdk1.7
#export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:$PATH
export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:/home/hadoop/hadoop2/sbin:$PATH
-
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

hadoop2/sbin 디렉토리에서 vim 편집기로 /home/hadoop/.bashrc 내용 편집

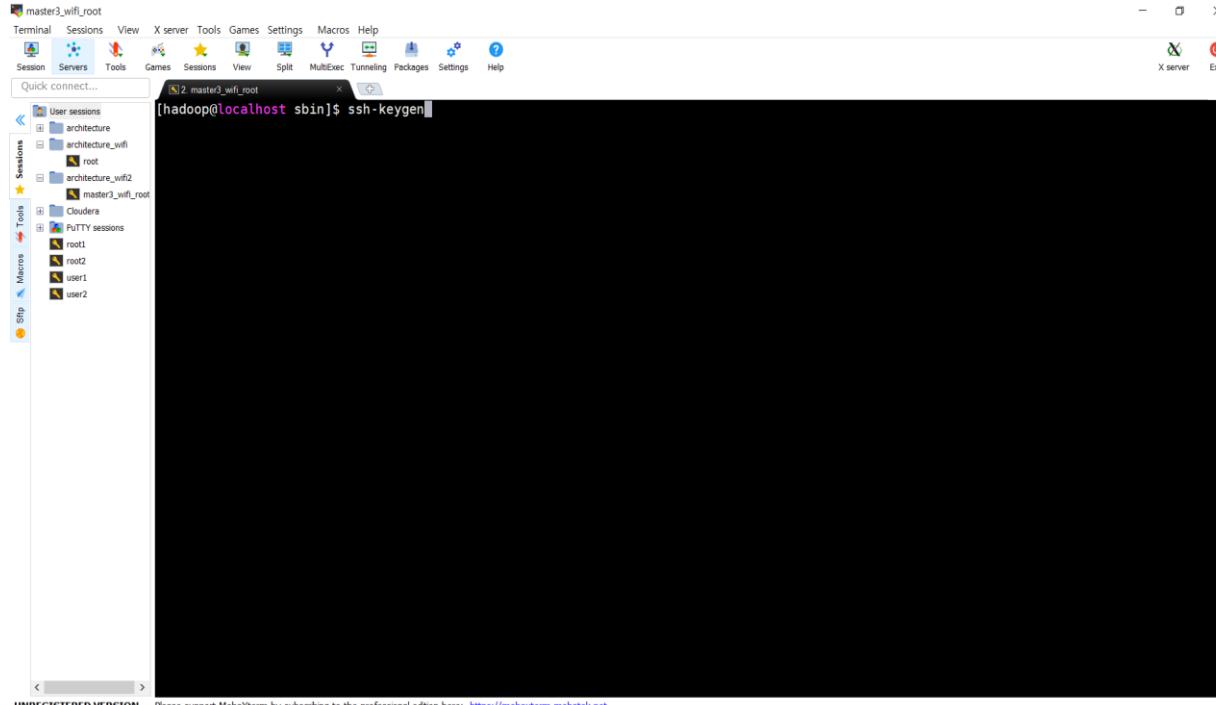
[hadoop@localhost sbin]\$ vim /home/hadoop/.bashrc

export PATH=\$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:\$PATH 를

export

PATH=\$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:/home/hadoop/hadoop2/sbin:\$PATH로 변경

가상 분산 모드

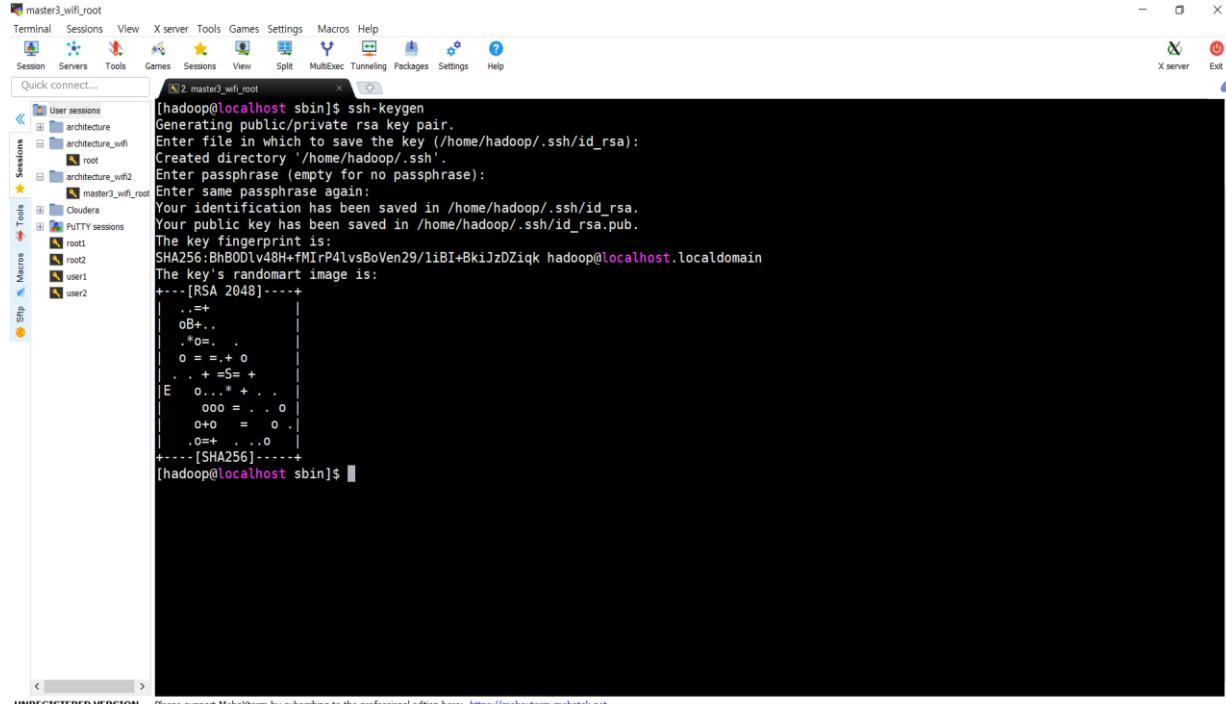


hadoop2/sbin 디렉토리 위치에서
hadoop 계정의 ssh key 등록 명령어 실행

[hadoop@localhost sbin]\$ ssh-keygen

Ssh는 서버 간 통신을 할때 ssh key를 통해 통신!!!!!!

가상 분산 모드



```
[hadoop@localhost sbin]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
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:BhBDLv48H+fIrPAlvsBoVen29/1iBI+BkIjzDZiqk hadoop@localhost.localdomain
The key's randomart image is:
+---[RSA 2048]---+
| ..+= |
| .B+.. |
| .*o=. . |
| 0 = =.+ 0 |
| . . + =S= + . |
| E 0 ...* + . . |
| 000 = . . 0 |
| 0+0 = 0 . |
| .o=+ . . 0 |
+---[SHA256]---+
[hadoop@localhost sbin]$
```

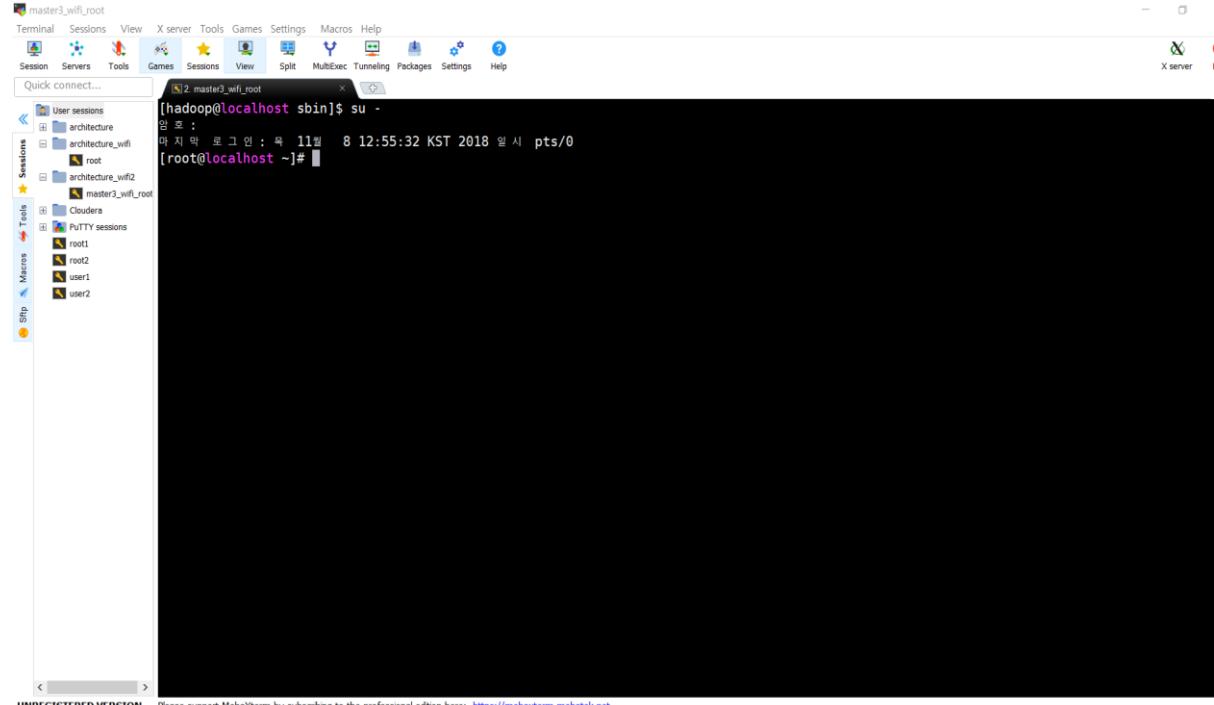
hadoop2/sbin 디렉토리 위치에서
hadoop 계정의 ssh key 등록 명령어 실행

[hadoop@localhost sbin]\$ ssh-keygen

Ssh는 서버 간 통신을 할때 ssh key를 통해 통신!!!!!!

생성할 때 총 3번의 enter key를 눌러야함. 아무 입력 없이 3번 enter key 누르기!!!!

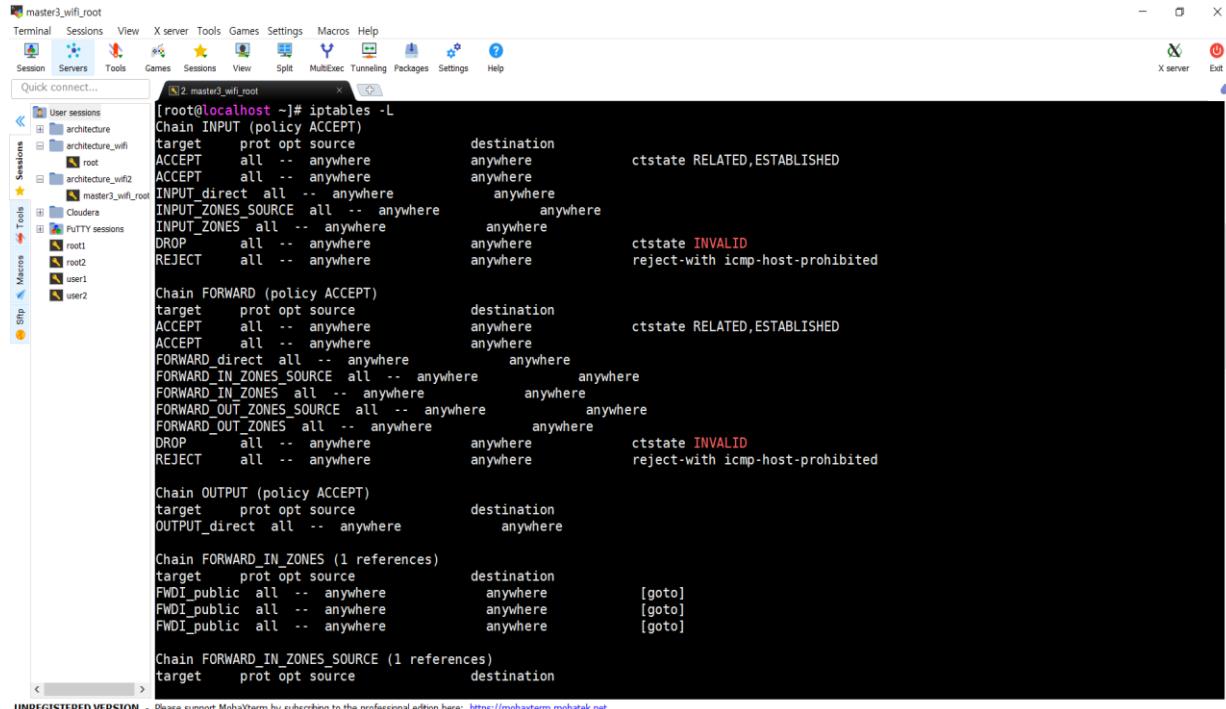
가상 분산 모드



hadoop2/sbin 디렉토리 위치에서
root 계정로 변경 후 방화벽 확인 및 종료

```
[hadoop@localhost sbin]$ su -
[root@localhost ~]#
```

가상 분산 모드



The screenshot shows a terminal window in MobaXterm with the title 'master3_wifi_root'. The command 'iptables -L' is run, displaying the current iptables rules. The output is as follows:

```
[root@localhost ~]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
ACCEPT    all  --  anywhere        anywhere        ctstate RELATED,ESTABLISHED
ACCEPT    all  --  anywhere        anywhere        anywhere
INPUT_direct all  --  anywhere        anywhere        anywhere
INPUT_ZONES_SOURCE all  --  anywhere        anywhere        anywhere
INPUT_ZONES all  --  anywhere        anywhere        anywhere
DROP      all  --  anywhere        anywhere        ctstate INVALID
REJECT   all  --  anywhere        anywhere        reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source          destination
ACCEPT    all  --  anywhere        anywhere        ctstate RELATED,ESTABLISHED
ACCEPT    all  --  anywhere        anywhere        anywhere
FORWARD_direct all  --  anywhere        anywhere        anywhere
FORWARD_IN_ZONES_SOURCE all  --  anywhere        anywhere        anywhere
FORWARD_IN_ZONES all  --  anywhere        anywhere        anywhere
FORWARD_OUT_ZONES_SOURCE all  --  anywhere        anywhere        anywhere
FORWARD_OUT_ZONES all  --  anywhere        anywhere        anywhere
DROP      all  --  anywhere        anywhere        ctstate INVALID
REJECT   all  --  anywhere        anywhere        reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
OUTPUT_direct all  --  anywhere        anywhere

Chain FORWARD_IN_ZONES (1 references)
target     prot opt source          destination
FWDI_public all  --  anywhere        anywhere        [goto]
FWDI_public all  --  anywhere        anywhere        [goto]
FWDI_public all  --  anywhere        anywhere        [goto]

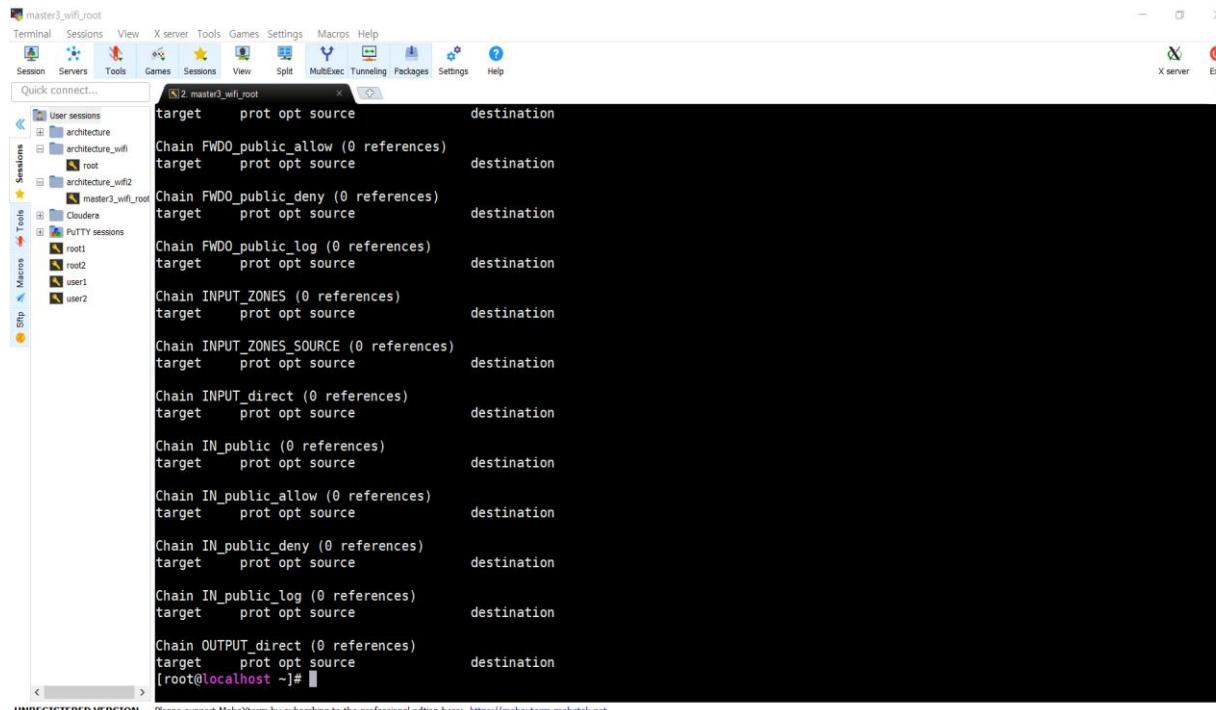
Chain FORWARD_IN_ZONES_SOURCE (1 references)
target     prot opt source          destination
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

hadoop2/sbin 디렉토리 위치에서
root 계정로 변경 후 방화벽 확인 및 종료

```
[root@localhost ~]# iptables -L
[root@localhost ~]# iptables -F
[root@localhost ~]# iptables -L
```

가상 분산 모드



The screenshot shows a MobaXterm window with a terminal session titled "master3_wifi_root". The session list on the left shows "User sessions" like "architecture", "architecture_wifi", and "PUTTY sessions" for users "root1", "root2", "user1", and "user2". The terminal window displays the output of the "iptables -L" command, listing various chains and their rules:

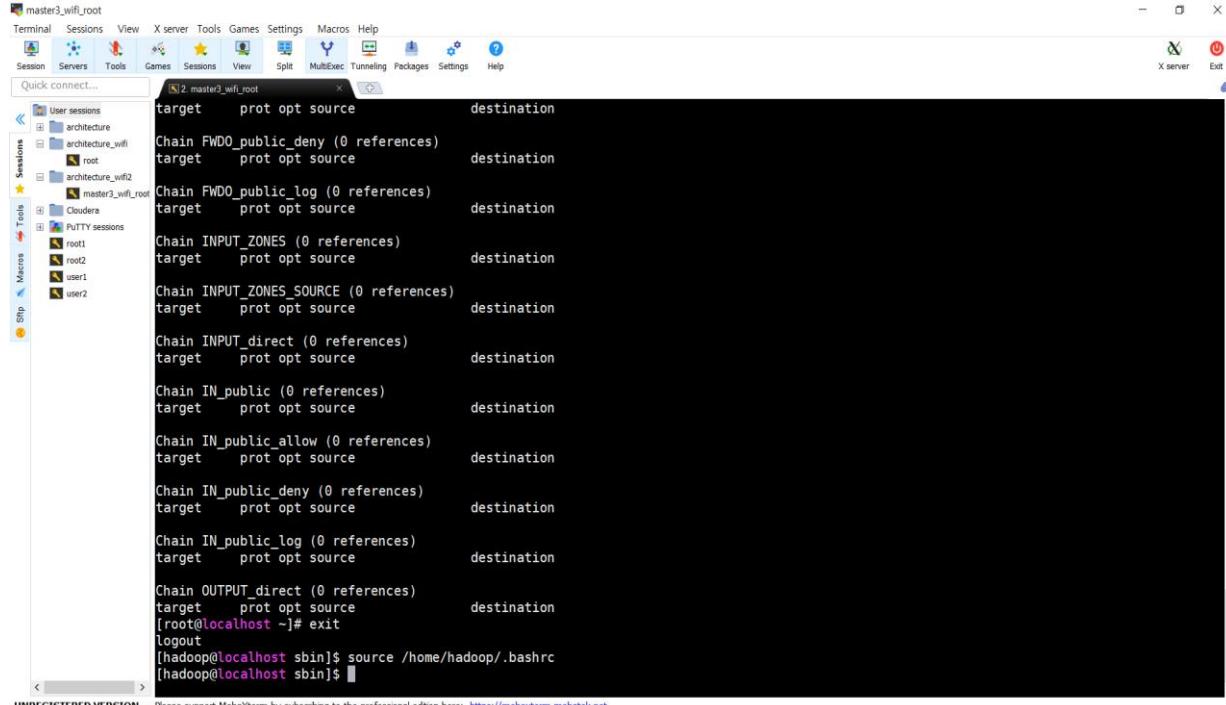
```
target  prot opt source          destination
Chain FWDO_public_allow (0 references)
      target  prot opt source          destination
Chain FWDO_public_deny (0 references)
      target  prot opt source          destination
Chain FWDO_public_log (0 references)
      target  prot opt source          destination
Chain INPUT_ZONES (0 references)
      target  prot opt source          destination
Chain INPUT_ZONES_SOURCE (0 references)
      target  prot opt source          destination
Chain INPUT_direct (0 references)
      target  prot opt source          destination
Chain IN_public (0 references)
      target  prot opt source          destination
Chain IN_public_allow (0 references)
      target  prot opt source          destination
Chain IN_public_deny (0 references)
      target  prot opt source          destination
Chain IN_public_log (0 references)
      target  prot opt source          destination
Chain OUTPUT_direct (0 references)
      target  prot opt source          destination
[root@localhost ~]#
```

At the bottom of the terminal window, there is a footer message: "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

hadoop2/sbin 디렉토리 위치에서
root 계정로 변경 후 방화벽 확인 및 종료

```
[root@localhost ~]# iptables -L
[root@loalchost ~]# iptables -F
[root@localhost ~]# iptables -L
```

가상 분산 모드



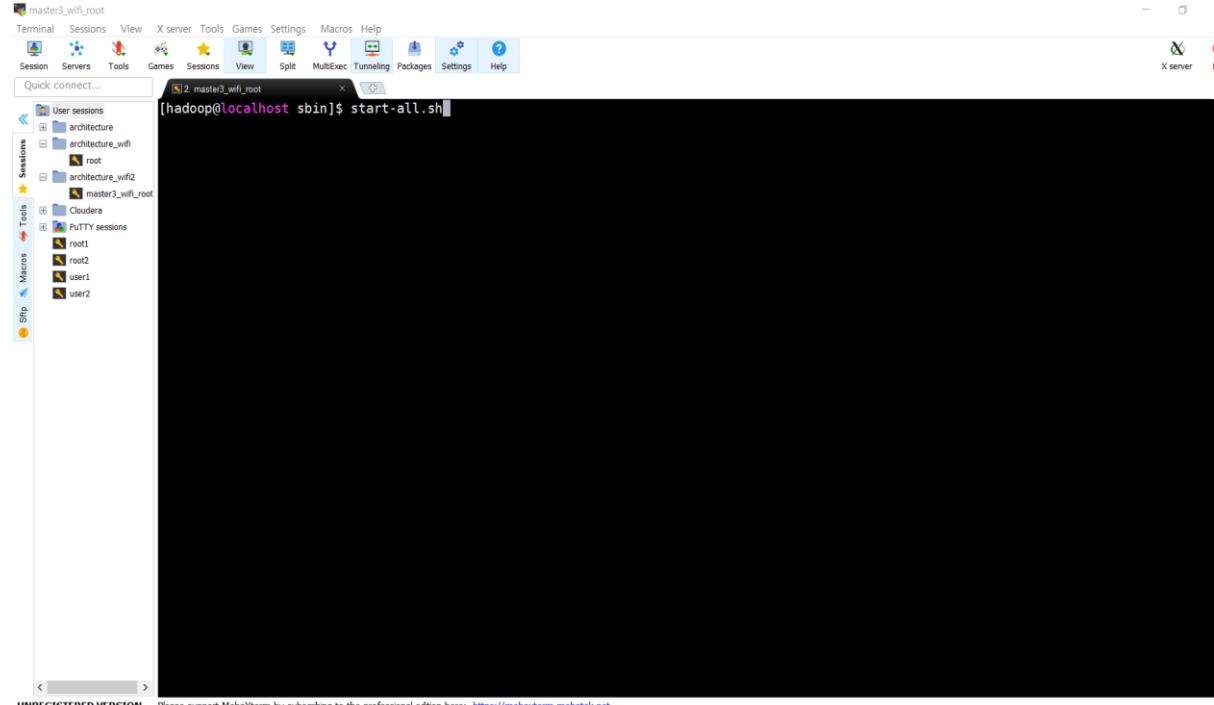
```
target      prot opt source          destination
Chain FWDO_public_deny (0 references)
target      prot opt source          destination
Chain FWDO_public_log (0 references)
target      prot opt source          destination
Chain INPUT_ZONES (0 references)
target      prot opt source          destination
Chain INPUT_ZONES_SOURCE (0 references)
target      prot opt source          destination
Chain INPUT_direct (0 references)
target      prot opt source          destination
Chain IN_public (0 references)
target      prot opt source          destination
Chain IN_public_allow (0 references)
target      prot opt source          destination
Chain IN_public_deny (0 references)
target      prot opt source          destination
Chain IN_public_log (0 references)
target      prot opt source          destination
Chain OUTPUT_direct (0 references)
target      prot opt source          destination
[root@localhost ~]# exit
[unregisterd@localhost ~]$ source /home/hadoop/.bashrc
[hadoop@localhost ~]$
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

root 계정 exit 한 뒤 /home/hadoop/.bashrc를 source로 실행

```
[root@localhost ~]# exit
[hadoop@localhost sbin]$ source /home/hadoop/.bashrc
```

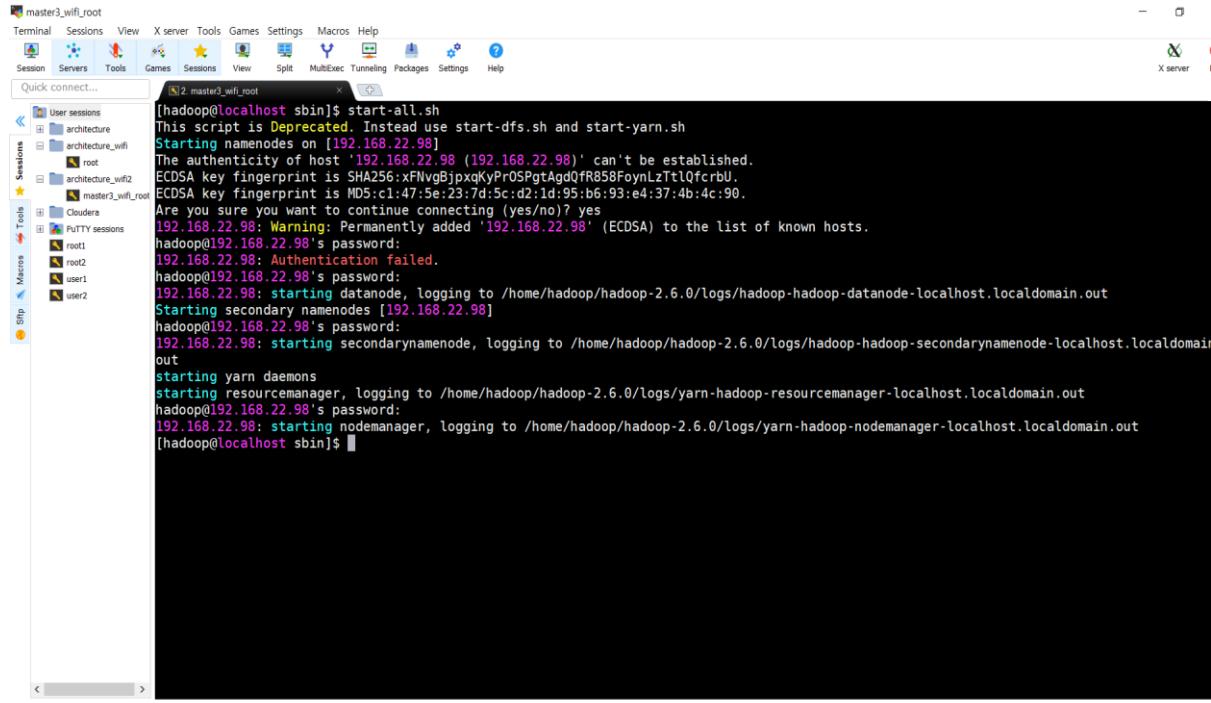
가상 분산 모드



Start-all.sh 실행

[hadoop@localhost sbin]\$ start-all.sh

가상 분산 모드



The screenshot shows a terminal window in MobaXterm titled 'master3_wifi_root'. The command 'start-all.sh' is being run on the localhost. The output indicates that the script is deprecated and instead should use 'start-dfs.sh' and 'start-yarn.sh'. It shows the authentication process for the host '192.168.22.98', including ECDSA key fingerprints and MD5 checksums. A warning message states that the host has been permanently added to the list of known hosts. The command then starts the datanode, secondary namenodes, yarn daemons, resource manager, and nodemanager services. The session is running under the 'hadoop' user.

```
[hadoop@localhost sbin]$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [192.168.22.98]
The authenticity of host '192.168.22.98 (192.168.22.98)' can't be established.
ECDSA key fingerprint is SHA256:xNvgBjpxKyPr0SPgtAgd0fR858FoynLzTtlofcrbU.
ECDSA key fingerprint is MD5:c1:47:5e:23:7d:5c:d2:id:95:b6:93:e4:37:4b:4c:90.
Are you sure you want to continue connecting (yes/no)? yes
192.168.22.98: Warning: Permanently added '192.168.22.98' (ECDSA) to the list of known hosts.
hadoop@192.168.22.98's password:
192.168.22.98: Authentication failed.
hadoop@192.168.22.98's password:
192.168.22.98: starting datanode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hadoop-datanode-localhost.localdomain.out
Starting secondary namenodes [192.168.22.98]
hadoop@192.168.22.98's password:
192.168.22.98: starting secondarynamenode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hadoop-secondarynamenode-localhost.localdomain.out
starting yarn daemons
starting resourcemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-hadoop-resourcemanager-localhost.localdomain.out
hadoop@192.168.22.98's password:
192.168.22.98: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-hadoop-nodemanager-localhost.localdomain.out
[hadoop@localhost sbin]$
```

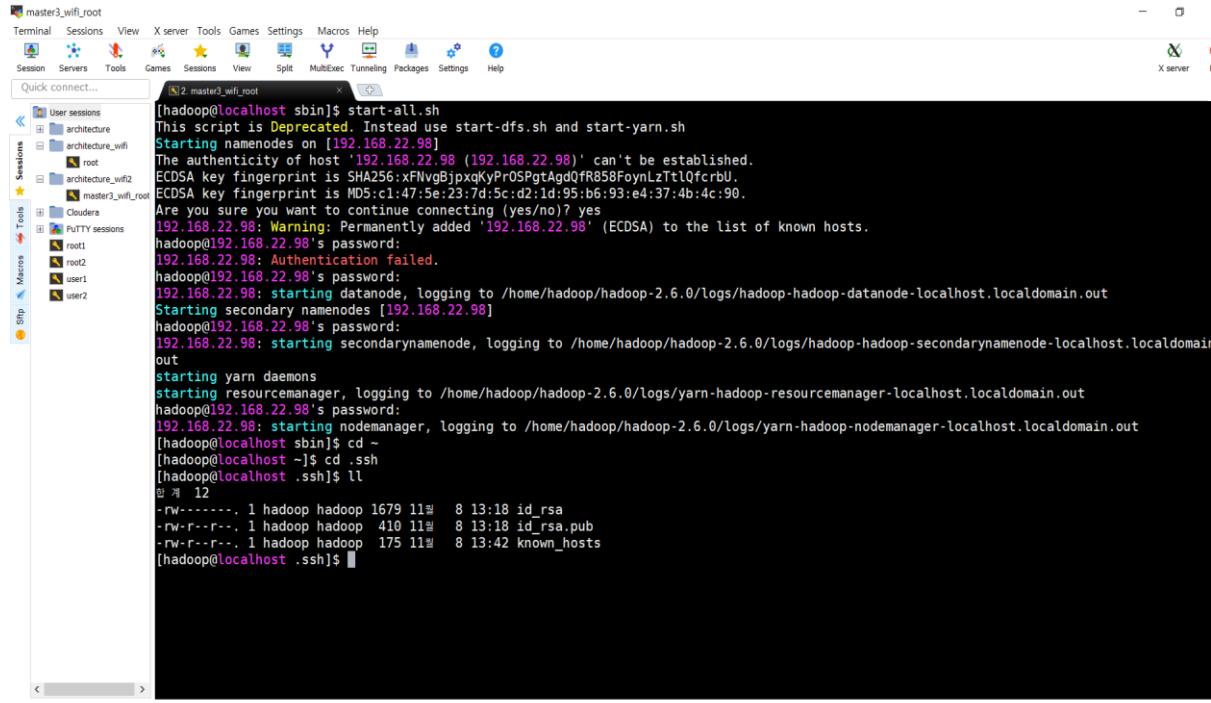
Start-all.sh 실행

[hadoop@localhost sbin]\$ start-all.sh

Are you sure you want to continue connecting (yes/no)? Yes 입력

단계적으로 hadoop 계정의 비밀번호 입력하기....

가상 분산 모드



```
[hadoop@localhost sbin]$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [192.168.22.98]
The authenticity of host '192.168.22.98 (192.168.22.98)' can't be established.
ECDSA key fingerprint is SHA256:xNvgBjpxqKyPr0SPgtAgd0fR858FoynLzTtlofcrbU.
ECDSA key fingerprint is MD5:c1:47:5e:23:7d:5c:d2:id:95:b6:93:e4:37:4b:4c:90.
Are you sure you want to continue connecting (yes/no)? yes
192.168.22.98: Warning: Permanently added "192.168.22.98" (ECDSA) to the list of known hosts.
hadoop@192.168.22.98's password:
192.168.22.98: Authentication failed.
hadoop@192.168.22.98's password:
192.168.22.98: starting datanode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hadoop-datanode-localhost.localdomain.out
Starting secondary namenodes [192.168.22.98]
hadoop@192.168.22.98's password:
192.168.22.98: starting secondarynamenode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hadoop-secondarynamenode-localhost.localdomain.out
starting yarn daemons
starting resourcemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-hadoop-resourcemanager-localhost.localdomain.out
hadoop@192.168.22.98's password:
192.168.22.98: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-hadoop-nodemanager-localhost.localdomain.out
[hadoop@localhost sbin]$ cd ~
[hadoop@localhost ~]$ cd .ssh
[hadoop@localhost .ssh]$ ll
hadoop@localhost .ssh]$ ll
合계 12
-rw-----. 1 hadoop hadoop 1679 11월 8 13:18 id_rsa
-rw-r--r--. 1 hadoop hadoop 410 11월 8 13:18 id_rsa.pub
-rw-r--r--. 1 hadoop hadoop 175 11월 8 13:42 known_hosts
[hadoop@localhost .ssh]$
```

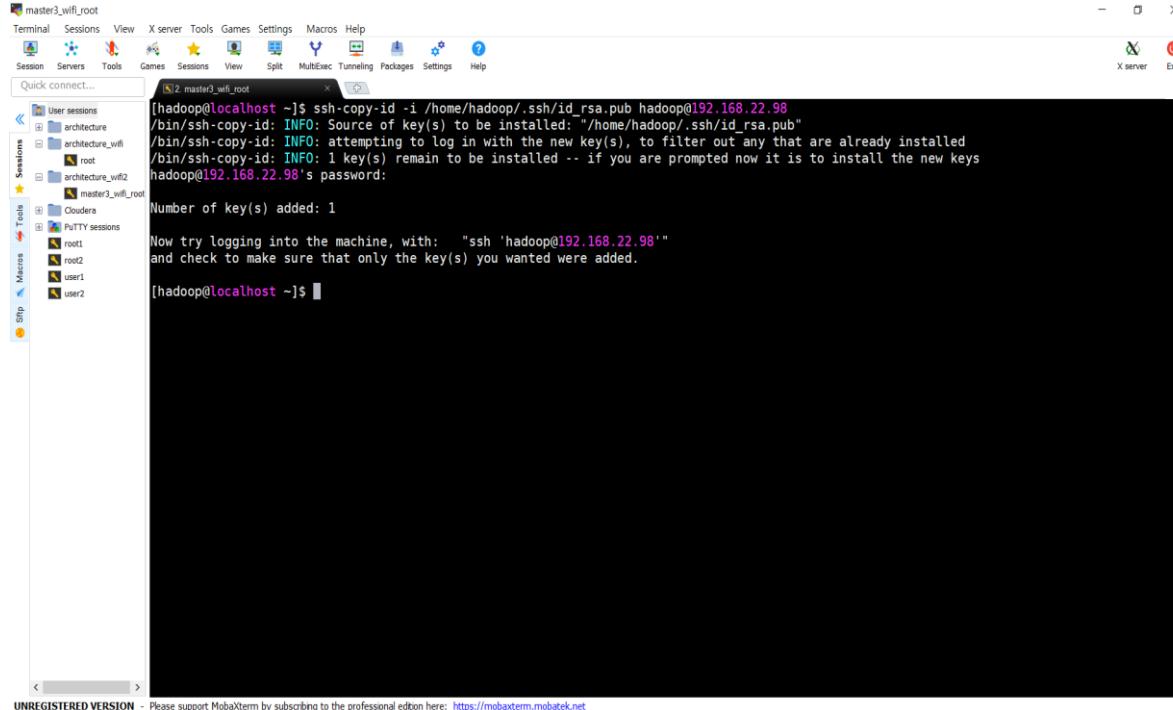
.ssh 디렉토리로 이동한 뒤 key 확인

```
[hadoop@localhost sbin]$ cd ~
[hadoop@localhost ~]$ cd .ssh
[hadoop@localhost .ssh]$ ll
```

rsa 키 확인한 후 상위 디렉토리로 이동

```
[hadoop@localhost .ssh]$ cd ..
```

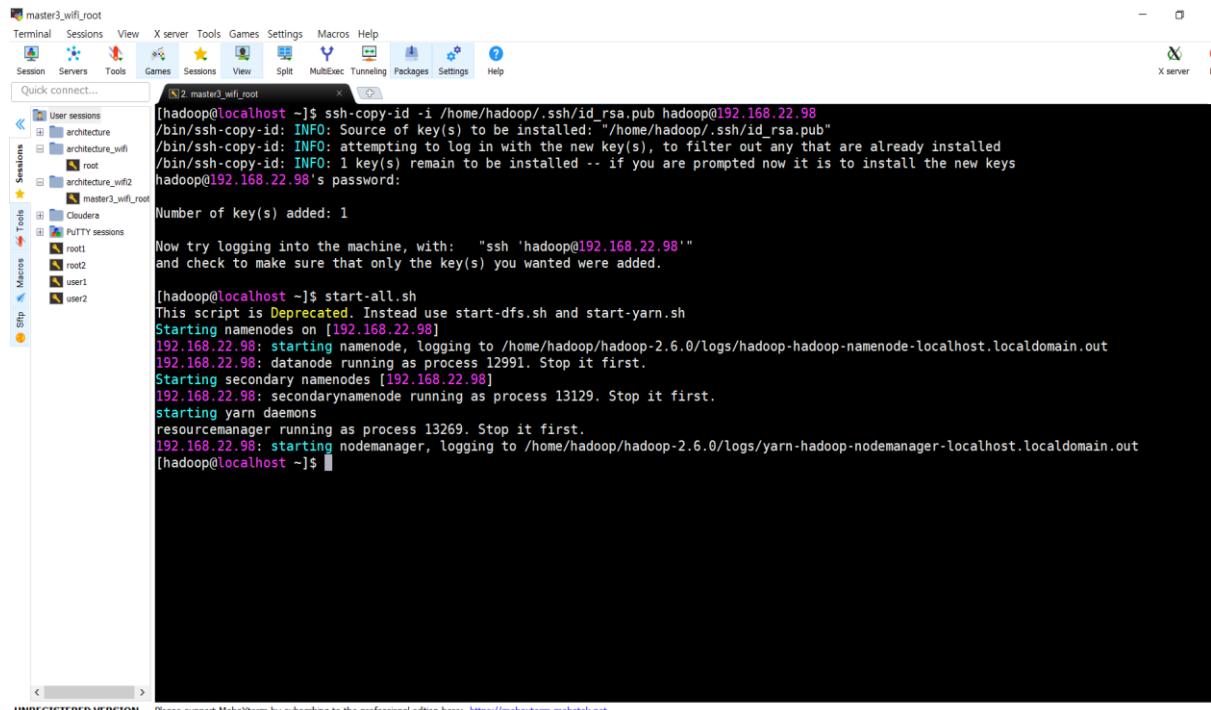
가상 분산 모드



id_rsa.pub key를 hadoop 계정으로 복사

[hadoop@localhost ~]\$ ssh-copy-id -i /home/hadoop/.ssh/id_rsa.pub [hadoop@192.168.22.98](https://192.168.22.98)
hadoop 계정의 password 입력

가상 분산 모드



```
[hadoop@localhost ~]$ ssh-copy-id -i /home/hadoop/.ssh/id_rsa.pub hadoop@192.168.22.98
/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hadoop/.ssh/id_rsa.pub"
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
hadoop@192.168.22.98's password:
Number of key(s) added: 1

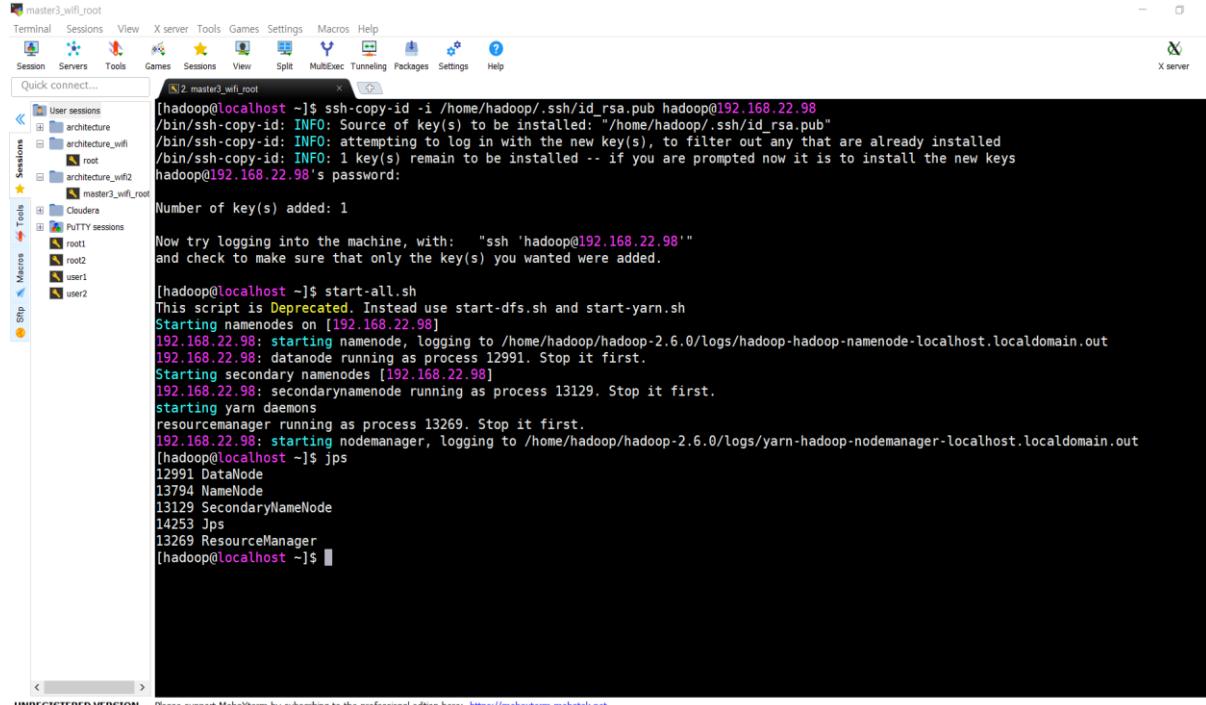
Now try logging into the machine, with: "ssh 'hadoop@192.168.22.98'"
and check to make sure that only the key(s) you wanted were added.

[hadoop@localhost ~]$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [192.168.22.98]
192.168.22.98: starting namenode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hadoop-namenode-localhost.localdomain.out
192.168.22.98: datanode running as process 12991. Stop it first.
Starting secondary namenodes [192.168.22.98]
192.168.22.98: secondarynamenode running as process 13129. Stop it first.
starting yarn daemons
resourcemanager running as process 13269. Stop it first.
192.168.22.98: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-hadoop-nodemanager-localhost.localdomain.out
[hadoop@localhost ~]$
```

Start-all.sh 실행

```
[hadoop@localhost ~]$ start-all.sh
```

가상 분산 모드



The screenshot shows a MobaXterm window titled "master3_wifi_root". The terminal session displays the following commands and output:

```
[hadoop@localhost ~]$ ssh-copy-id -i /home/hadoop/.ssh/id_rsa.pub hadoop@192.168.22.98
/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hadoop/.ssh/id_rsa.pub"
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
hadoop@192.168.22.98's password:
Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'hadoop@192.168.22.98'"
and check to make sure that only the key(s) you wanted were added.

[hadoop@localhost ~]$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [192.168.22.98]
192.168.22.98: starting namenode, logging to /home/hadoop/hadoop-2.6.0/logs/hadoop-hadoop-namenode-localhost.localdomain.out
192.168.22.98: datanode running as process 12991. Stop it first.
Starting secondary namenodes [192.168.22.98]
192.168.22.98: secondarynamenode running as process 13129. Stop it first.
starting yarn daemons
resourcemanager running as process 13269. Stop it first.
192.168.22.98: starting nodemanager, logging to /home/hadoop/hadoop-2.6.0/logs/yarn-hadoop-nodemanager-localhost.localdomain.out
[hadoop@localhost ~]$ jps
12991 DataNode
13794 NameNode
13129 SecondaryNameNode
14253 Jps
13269 ResourceManager
[hadoop@localhost ~]$
```

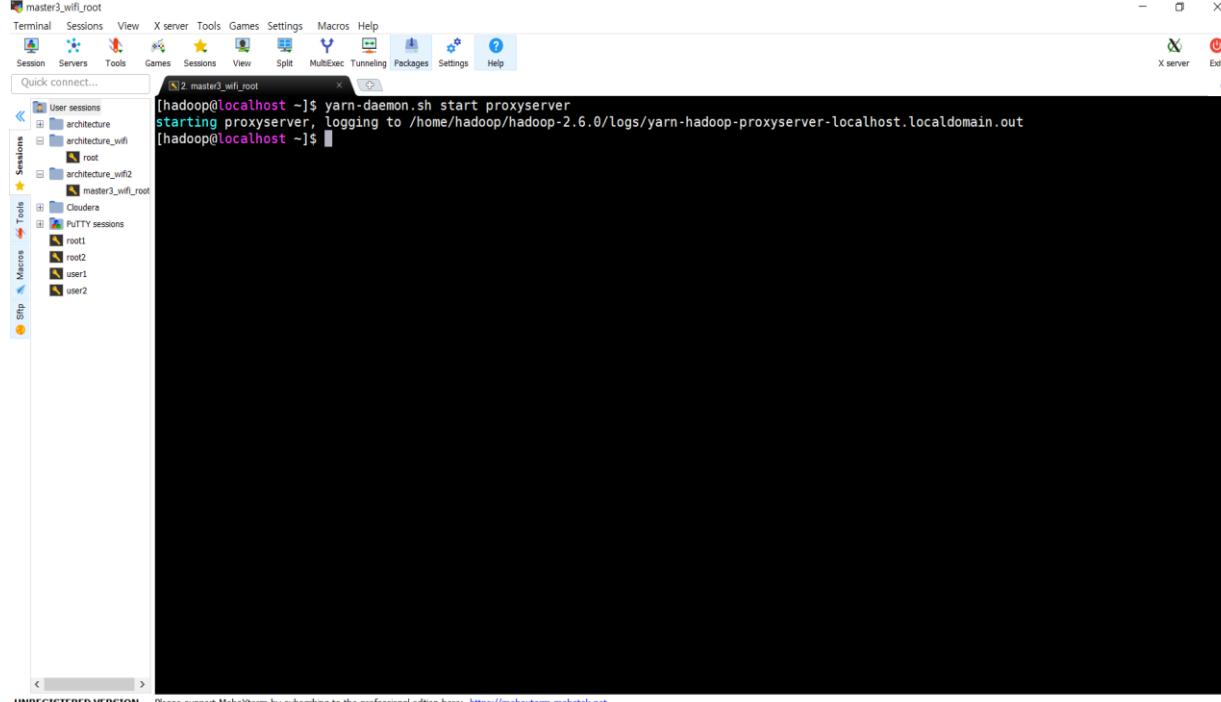
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

프로세스 세팅 확인

```
[hadoop@localhost ~]$ jps
```

프록시 서버 시작

[hadoop@localhost ~]\$ yarn-daemon.sh start proxyserver



가상 분산 모드

```
drwx----- 3 hadoop hadoop 4096 11월  8 20:51 datanode
drwxrwxr-x  3 hadoop hadoop 4096 11월  8 20:51 namenode
drwxrwxr-x  3 hadoop hadoop 4096 11월  8 20:51 namesecondary
[hadoop@team4 dfs]$ cd datanode
[hadoop@team4 datanode]$ ll
한 게 4
drwxrwxr-x  3 hadoop hadoop 4096 11월  7 23:57 current
[hadoop@team4 datanode]$ stop-all.sh
This script is Deprecated. Instead use stop-dfs.sh and stop-yarn.sh
Stopping namenodes on [team4]
team4: stopping namenode
192.168.100.107: no datanode to stop
Stopping secondary namenodes [team4]
team4: stopping secondarynamenode
stopping yarn daemons
stopping resourcemanager
192.168.100.107: stopping nodemanager
no proxyserver to stop
[hadoop@team4 datanode]$ rm -r datanode
rm: cannot remove `datanode': 그런 파일이나 디렉터리가 없습니다
[hadoop@team4 datanode]$ cd ..
[hadoop@team4 dfs]$ rm -r datanode
[hadoop@team4 dfs]$ ll
한 게 8
drwxrwxr-x  3 hadoop hadoop 4096 11월  8 20:53 namenode
drwxrwxr-x  3 hadoop hadoop 4096 11월  8 20:53 namesecondary
[hadoop@team4 dfs]$ mkdir datanode
[hadoop@team4 dfs]$ cd datanode
[hadoop@team4 datanode]$ hdfs namenode -format
18/11/08 20:54:40 INFO namenode.NameNode: STARTUP_MSG:
```

Namenode 미실행시

\$hdfs namenode -format를 통해 포맷하여 준다

Datanode 미실행시

/home/hadoop/data/dfs/datanode 를 삭제 후 hadoop을 재실행(start-dfs.sh , start-yarn.sh)

가상 분산 모드

```
[root@team4 etc]# vim hosts  
[root@team4 etc]#
```

```
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4  
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6  
192.168.100.107 team4  
192.168.100.108 Master  
192.168.100.104 slave01  
192.168.100.105 slave02
```

주키퍼설치를 위한 사전 작업

1. Cluser를 구성하고자 하는 server 중 3대의 server Root 계정에서 user, pw 설정
`#adduser zookeeper
#passwd zookeeper`
 이후 비밀번호 설정
2. Host에서 클러스터를 구성하고자 하는 가상머신의 호스트 네임과 ip 주소를 등록한다.

가상 분산 모드

```
[zookeeper@team4 ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/zookeeper/.ssh/id_rsa):
Created directory '/home/zookeeper/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/zookeeper/.ssh/id_rsa.
Your public key has been saved in /home/zookeeper/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Iwkklna3WfBr3b8MFUdc+ciLlNsamtBbY5IeX6e0uA zookeeper@team4
The key's randomart image is:
+---[RSA 2048]---+
| ..oo   . .+o |
| oo     oo.oB |
| .. . .o .o+o+o |
| . . . o .Xo+o |
| o . . o So = *o |
| . . . .o . E o. |
|   o oo . |
|     o oo |
+---[SHA256]-----+
```

```
[zookeeper@team4 ~]$ ssh-copy-id -i .ssh/id_rsa.pub zookeeper@192.168.100.107
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zookeeper/.ssh/id_rsa.pub"
The authenticity of host '192.168.100.107 (192.168.100.107)' can't be established.
ECDSA key fingerprint is SHA256:MKZIPwu483V7gXT3c8EN086BNh8Wk6bksvk2jxUko.
ECDSA key fingerprint is MD5:a2:d0:75:55:ab:b5:ad:62:34:30:b6:43:e2:86:41:b1.
Are you sure you want to continue connecting (yes/no)? 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 prompted now it is to install all the new keys
zookeeper@192.168.100.107's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'zookeeper@192.168.100.107'"
and check to make sure that only the key(s) you wanted were added.
```

```
[zookeeper@team4 ~]$ ssh-copy-id -i .ssh/id_rsa.pub zookeeper@192.168.100.108
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zookeeper/.ssh/id_rsa.pub"
The authenticity of host '192.168.100.108 (192.168.100.108)' can't be established.
ECDSA key fingerprint is SHA256:aC+8toxSzWZxjsUfMoZSBXZd36eBupTW2qxLNh+DTZo.
ECDSA key fingerprint is MD5:f1:be:4e:87:ef:64:1c:7a:de:b8:43:8b:c:f:6f:b4:30.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
```

```
[zookeeper@team4 ~]$ ssh zookeeper@192.168.100.108
[zookeeper@master ~]$ exit
logout
Connection to 192.168.100.108 closed.
[zookeeper@team4 ~]$
```

- 각 서버의 zookeeper 계정에서 ssh-key 값을 설정한다.

```
$ssh-keygen
```

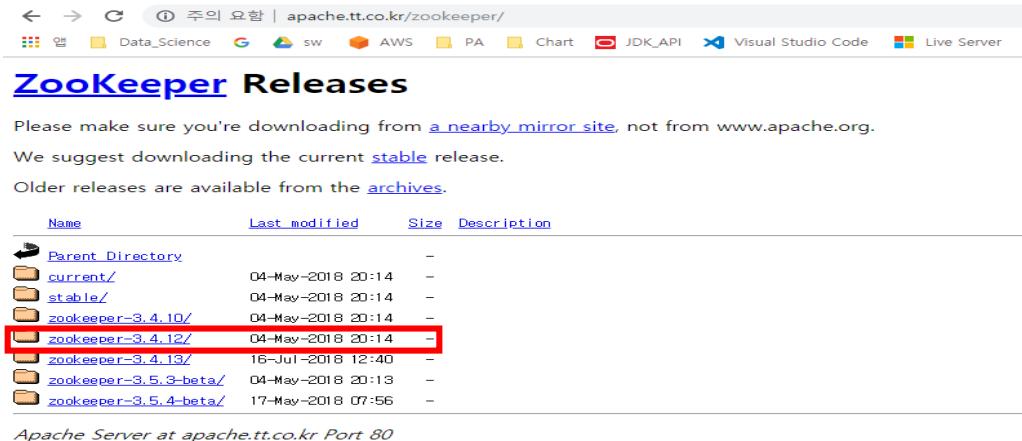
:이 때 '엔터'를 연속으로 클릭하여 비밀 번호를 설정하지 않는다.

- 서버간에 ssh-key값을 공유한다.

```
$ssh-copy-id -i /home/zookeeper/.ssh/id_rsa.pub zookeeper@ip주소
```

- Ssh zookeeper@ip주소를 통해 접속여부를 확인한다.

가상 분산 모드



Please make sure you're downloading from [a nearby mirror site](#), not from www.apache.org.
We suggest downloading the current [stable](#) release.
Older releases are available from the [archives](#).

| Name | Last modified | Size | Description |
|--------------------------|-------------------|------|-------------|
| Parent Directory | - | - | |
| current/ | 04-May-2018 20:14 | - | |
| stable/ | 04-May-2018 20:14 | - | |
| zookeeper-3.4.10/ | 04-May-2018 20:14 | - | |
| zookeeper-3.4.12/ | 04-May-2018 20:14 | - | |
| zookeeper-3.4.13/ | 16-Jul-2018 12:40 | - | |
| zookeeper-3.5.3-beta/ | 04-May-2018 20:13 | - | |
| zookeeper-3.5.4-beta/ | 17-May-2018 07:56 | - | |

Apache Server at apache.tt.co.kr Port 80

Index of /zookeeper/stable



| Name | Last modified | Size | Description |
|--------------------------------|-------------------|------|-------------|
| Parent Directory | - | - | |
| zookeeper-3.4.12.tar.gz | 26-Apr-2018 00:47 | 35M | |

Apache Server at apache.tt.co.kr Port 80

```
[zookeeper@team4 ~]$ wget http://apache.tt.co.kr/zookeeper/stable/zookeeper-3.4.12.tar.gz
--2018-11-08 22:00:42-- http://apache.tt.co.kr/zookeeper/stable/zookeeper-3.4.12.tar.gz
Resolving apache.tt.co.kr (apache.tt.co.kr) ... 211.47.69.77
Connecting to apache.tt.co.kr (apache.tt.co.kr)|211.47.69.77|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 36667596 (35M) [application/x-gzip]
Saving to: 'zookeeper-3.4.12.tar.gz'

100%[=====] 36,667,596  2.46MB/s   in 9.6s

2018-11-08 22:06:51 (3.65 MB/s) - 'zookeeper-3.4.12.tar.gz' saved [36667596/36667596]

[zookeeper@team4 ~]$ ll
合계 35812
-rw-rw-r-- 1 zookeeper zookeeper 36667596 26 Nov 2018 zookeeper-3.4.12.tar.gz
[zookeeper@team4 ~]$ tar xvzf zookeeper-3.4.12.tar.gz
zookeeper-3.4.12/
zookeeper-3.4.12/contrib/
zookeeper-3.4.12/contrib/rest/
zookeeper-3.4.12/contrib/rest/lib/
zookeeper-3.4.12/contrib/rest/lib/slf4j-log4j12-1.6.1.jar
zookeeper-3.4.12/contrib/rest/lib/jackson-core-asl-1.1.1.jar
zookeeper-3.4.12/contrib/rest/lib/jsr311-api-1.1.1.jar
zookeeper-3.4.12/contrib/rest/lib/jettison-1.1.jar
```

6. 주기퍼 다운로드 및 압축을 풀어준다.

- <http://apache.tt.co.kr/zookeeper/> 사이트에서 zookeeper-3.4.12.tar.gz 확인
- wget <http://apache.tt.co.kr/zookeeper/stable/zookeeper-3.4.12.tar.gz>
- tar xvzf zookeeper-3.4.12.gz

가상 분산 모드

```
drwxr-xr-x. 10 zookeeper zookeeper 4096 3월 27 2018 zookeeper-3.4.12
-rw-rw-r--. 1 zookeeper zookeeper 36667596 4월 26 2018 zookeeper-3.4.12.tar.gz
[zookeeper@team4 ~]$ cd zookeeper-3.4.12
[zookeeper@team4 zookeeper-3.4.12]$ ll
합계 1624
-rw-rw-r--. 1 zookeeper zookeeper 11938 3월 27 2018 LICENSE.txt
-rw-rw-r--. 1 zookeeper zookeeper 3132 3월 27 2018 NOTICE.txt
-rw-rw-r--. 1 zookeeper zookeeper 1585 3월 27 2018 README.md
-rw-rw-r--. 1 zookeeper zookeeper 1770 3월 27 2018 README_packaging.txt
drwxr-xr-x. 2 zookeeper zookeeper 4096 3월 27 2018 bin
-rw-rw-r--. 1 zookeeper zookeeper 87945 3월 27 2018 build.xml
drwxr-xr-x. 2 zookeeper zookeeper 4096 3월 27 2018 conf
drwxr-xr-x. 10 zookeeper zookeeper 4096 3월 27 2018 contrib
drwxr-xr-x. 2 zookeeper zookeeper 4096 3월 27 2018 dist-maven
drwxr-xr-x. 6 zookeeper zookeeper 4096 3월 27 2018 docs
-rw-rw-r--. 1 zookeeper zookeeper 8197 3월 27 2018 ivy.xml
-rw-rw-r--. 1 zookeeper zookeeper 1709 3월 27 2018 ivysettings.xml
drwxr-xr-x. 4 zookeeper zookeeper 4096 3월 27 2018 lib
drwxr-xr-x. 5 zookeeper zookeeper 4096 3월 27 2018 recipes
drwxr-xr-x. 8 zookeeper zookeeper 4096 3월 27 2018 src
-rw-rw-r--. 1 zookeeper zookeeper 1483366 3월 27 2018 zookeeper-3.4.12.jar
-rw-rw-r--. 1 zookeeper zookeeper 819 3월 27 2018 zookeeper-3.4.12.jar.asc
```

7. 간편한 사용을 위하여 명칭을 변경하여 준다. (별도의 캡처사진은 없음)

```
$ln -s zookeeper-3.4.12 zookeeper3
```

8. /home/zookeeper/zookeeper3/conf 의 zoo_sample.cfg를 zoo.cfg로 변경한다.

```
$cp zoo_sample.cfg zoo.cfg
```

```
-rw-rw-r--. 1 zookeeper zookeeper 535 3월 27 2018 configuration.xsl
-rw-rw-r--. 1 zookeeper zookeeper 2161 3월 27 2018 log4j.properties
-rw-rw-r--. 1 zookeeper zookeeper 922 3월 27 2018 zoo_sample.cfg
[zookeeper@team4 conf]$ cp zoo_sample.cfg zoo.cfg
[zookeeper@team4 conf]$ ll
합계 16
-rw-rw-r--. 1 zookeeper zookeeper 535 3월 27 2018 configuration.xsl
-rw-rw-r--. 1 zookeeper zookeeper 2161 3월 27 2018 log4j.properties
-rw-rw-r--. 1 zookeeper zookeeper 922 11월 8 22:12 zoo.cfg
-rw-rw-r--. 1 zookeeper zookeeper 922 3월 27 2018 zoo_sample.cfg
[zookeeper@team4 conf]$ vi zoo.cfg
[zookeeper@team4 conf]$
```

가상 분산 모드

```
#dataDir=/tmp/zookeeper  
dataDir=/home/zookeeper/data  
# the port at which the clients will connect  
clientPort=2181  
# the maximum number of client connections.  
# increase this if you need to handle more clients  
maxClientCnxns=0  
maxSessionTimeout=180000  
server.1=team4:2888:3888  
server.2=Master:2888:3888  
server.3=slave01:2888:3888
```

```
[zookeeper@master ~]$ cd ..  
[zookeeper@master ~]$ ls  
data  zookeeper-3.4.12  zookeeper-3.4.12.tar.gz  zookeeper.out  zookeeper3  
[zookeeper@master ~]$ cd data  
[zookeeper@master data]$ ls  
myid  version-2  zookeeper_server.pid  
[zookeeper@master data]$ vim myid
```

```
2  
~  
~
```

```
PATH=$PATH:$HOME/.local/bin:$HOME/bin:$ZOOKEEPER_HOME/bin:$JAVA_HOME/bin  
#PATH=$PATH:$HOME/bin:$ZOOKEEPER_HOME/bin:$JAVA_HOME/bin  
#PATH=$JAVA_HOME/bin:$ZOOKEEPER_HOME/bin:$PATH  
export PATH  
export ZOOKEEPER_HOME=/home/zookeeper/zookeeper3  
export JAVA_HOME=/usr/java/jdk1.7
```

7. Vim zoo.cfg를 통하여 zoo.cfg를 설정한다.

```
dataDir=/home/zookeeper/data  
maxClientCnxns=0(무한대를 의미한다)  
maxSessionTimeout=180000  
server.1=host_name:2888:3888  
server.2=host_name:2888:3888  
server.3=host_name:2888:3888
```

2888 : 각각의 통신하기 위한 포트 3888 : 마스터 선출을 위한 포트

8. /home/zookeeper의 하위디렉토리에 data라는 디렉토리를 생성하고(없을 경우) myid 파일을 생성한다.

```
$vim myid
```

9. 7번에서 생성한 server의 번호를 입력한다.

10. 경로를 설정한다.

```
$vim ~/.bashrc
```

```
PATH=$PATH:$HOME/.local/bin:$HOME/bin:$ZOOKEEPER_HOME/bin:$JAVA_HOME/bin  
export ZOOKEEPER_HOME=/home/zookeeper/zookeeper3  
export JAVA_HOME=/usr/java/jdk1.7
```

완전 분산 모드로 진행
(No use Ambari)

완전 분산 모드

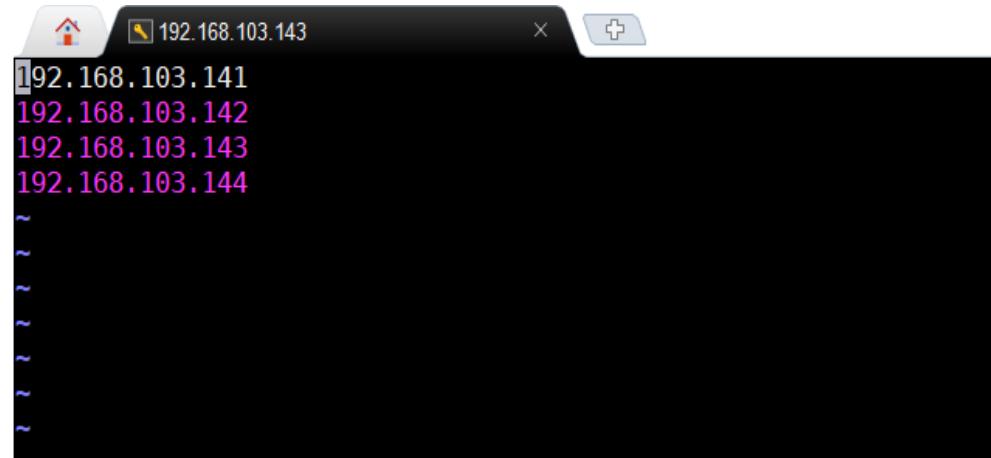
1. 가상분산모드로 설정된 가상컴퓨터에서 하둡만 삭제한다.

- java, Protocol Buffer는 삭제하지 않는다
- 다만, 이후 Spark, hive 설치시 버전 문제로 java를 업데이트해야 하이며 이는 해당 과정에서 설명 한다.

2. Namenode가 설정된 서버를 set up 하고 다른 서버에 배포한다.

3. 서버 설정 시 가상분산모드와 다른 설정만 제시하도록 한다.

완전 분산 모드



A screenshot of a terminal window titled "192.168.103.143". The window displays the following text:

```
192.168.103.141
192.168.103.142
192.168.103.143
192.168.103.144
~
~
~
~
~
~
~
```

hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로
slaves 파일 편집
파일 내 내용은 현재 cluster를 형성하는 호스트ip 주소
입력

```
[hadoop@localhost hadoop]$ vim slaves
```

완전 분산 모드

```
<!-- Put site-specific property overrides in this file. -->  
  
<configuration>  
    <property>  
        <name>fs.defaultFS</name>  
        <value>hdfs://hadoop-cluster</value>  
    </property>  
    <property>  
        <name>ha.zookeeper.quorum</name>  
        <value>jmj:2181,sdh:2181,khk:2181</value>  
    </property>  
    <property>  
        <name>ipc.client.connect.max.retries</name>  
        <value>10</value>  
    </property>  
    <property>  
        <name>ipc.client.connect.retry.interval</name>  
        <value>5000</value>  
    </property>  
</configuration>
```

hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로
core-site.xml 파일 편집

```
[hadoop@localhost hadoop]$ vim core-site.xml
```

* <configuration></configuration> 부분에서 안 부분을 수정해야함

```
<configuration>  
    <property>  
        <name>fs.defaultFS</name>  
        <value>hdfs://hadoop-cluster</value>  
    </property>  
    <property>  
        <name>ha.zookeeper.quorum</name>  
        <value>localhost:2181,master:2181,slave1:2181</value>  
    </property>  
</configuration>
```

완전 분산 모드

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>/home/hadoop/data/dfs/namenode</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>/home/hadoop/data/dfs/datanode</value>
    </property>
    <property>
        <name>dfs.journalnode.edits.dir</name>
        <value>/home/hadoop/data/dfs/journalnode</value>
    </property>
    <property>
        <name>dfs.nameservices</name>
        <value>hadoop-cluster</value>
    </property>
    <property>
        <name>dfs.ha.namenodes.hadoop-cluster</name>
        <value>nn1,nn2</value>
    </property>
    <property>
        <name>dfs.namenode.rpc-address.hadoop-cluster.nn1</name>
        <value>jmj:8020</value>
    </property>
    <property>
        <name>dfs.namenode.rpc-address.hadoop-cluster.nn2</name>
        <value>sdh:8020</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hadoop-cluster.nn1</name>
        <value>jmj:50070</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hadoop-cluster.nn2</name>
        <value>sdh:50090</value>
    </property>
```

hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로 hdfs-site.xml 파일 편집(1)

[hadoop@localhost hadoop]\$ vim hdfs-site.xml

* <configuration></configuration> 부분에서 안 부분을 수정해야함

다음과 같음

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>/home/hadoop/data/dfs/namenode</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>/home/hadoop/data/dfs/datanode</value>
    </property>
    <property>
        <name>dfs.journalnode.edits.dir</name>
        <value>/home/hadoop/data/dfs/journalnode</value>
    </property>
    <property>
        <name>dfs.nameservices</name>
        <value>hadoop-cluster</value>
    </property>
    <property>
        <name>dfs.ha.namenodes.hadoop-cluster</name>
        <value>nn1,nn2</value>
    </property>
    <property>
        <name>dfs.namenode.rpc-address.hadoop-cluster.nn1</name>
        <value>jmj:8020</value>
    </property>
    <property>
        <name>dfs.namenode.rpc-address.hadoop-cluster.nn2</name>
        <value>sdh:8020</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hadoop-cluster.nn1</name>
        <value>jmj:50070</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hadoop-cluster.nn2</name>
        <value>sdh:50090</value>
    </property>
```

완전 분산 모드

```
<property>
    <name>dfs.namenode.shared.edits.dir</name>
    <value>qjournal://jmj:8485;sdh:8485;khk:8485/hadoop-cluster</value>
</property>
<property>
    <name>dfs.client.failover.proxy.provider.hadoop-cluster</name>
    <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>
<property>
    <name>dfs.ha.fencing.methods</name>
    <value>sshfence</value>
</property>
<property>
    <name>dfs.ha.fencing.ssh.private-key-files</name>
    <value>/home/hadoop/.ssh/id_rsa</value>
</property>
<property>
    <name>dfs.ha.automatic-failover.enabled</name>
    <value>true</value>
</property>
<property>
    <name>dfs.block.size</name>
    <value>33554432</value>
</property>
<property>
    <name>dfs.hosts.exclude</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/exclude_list</value>
</property>
<property>
    <name>mapred.hosts.exclude</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/exclude_list</value>
</property>
<property>
    <name>dfs.hosts</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/include_list</value>
</property>
</configuration>
```

hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기로
hdfs-site.xml 파일 편집(2)

[hadoop@localhost hadoop]\$ vim hdfs-site.xml

* <configuration></configuration> 부분에서 안 부분을 수정해야함

다음과 같음

```
<property>
    <name>dfs.namenode.shared.edits.dir</name>
    <value>qjournal://jmj:8485;sdh:8485;khk:8485/hadoop-cluster</value>
</property>
<property>
    <name>dfs.client.failover.proxy.provider.hadoop-cluster</name>
    <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>
<property>
    <name>dfs.ha.fencing.methods</name>
    <value>sshfence</value>
</property>
<property>
    <name>dfs.ha.fencing.ssh.private-key-files</name>
    <value>/home/hadoop/.ssh/id_rsa</value>
</property>
<property>
    <name>dfs.ha.automatic-failover.enabled</name>
    <value>true</value>
</property>
<property>
    <name>dfs.block.size</name>
    <value>33554432</value>
</property>
<property>
    <name>dfs.hosts.exclude</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/exclude_list</value>
</property>
<property>
    <name>mapred.hosts.exclude</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/exclude_list</value>
</property>
<property>
    <name>dfs.hosts</name>
    <value>/home/hadoop/hadoop2/etc/hadoop/include_list</value>
</property>
</configuration>
```

완전 분산 모드

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle,spark_shuffle</value>
    </property>
    <property>
        <name>yarn.nodemanager.aux-services.mapreduce_shuffle.class</name>
        <value>org.apache.hadoop.mapred.ShuffleHandler</value>
    </property>
    <property>
        <name>yarn.nodemanager.aux-services.spark_shuffle.class</name>
        <value>org.apache.spark.network.yarn.YarnShuffleService</value>
    </property>
    <property>
        <name>spark.yarn.shuffle.stopOnFailure</name>
        <value>false</value>
    </property>
    <property>
        <name>yarn.nodemanager.local-dirs</name>
        <value>/home/hadoop/yarn/nm-local-dir</value>
    </property>
    <property>
        <name>yarn.resourcemanager.fs.state-store.url</name>
        <value>/home/hadoop/data/yarn/system/rmstore</value>
    </property>
    <property>
        <name>yarn.resourcemanager.hostname</name>
        <value>jmj</value>
    </property>
    <property>
        <name>yarn.web-proxy.address</name>
        <value>0.0.0.0:8089</value>
    </property>
    <property>
        <name>yarn.nodemanager.recovery.enabled</name>
        <value>true</value>
    </property>
    <property>
        <name>yarn.nodemanager.address</name>
        <value>0.0.0.0:45454</value>
    </property>
    <property>
        <name>yarn.nodemanager.pmem-check-enabled</name>
        <value>false</value>
    </property>
    <property>
        <name>yarn.nodemanager.vmem-check-enabled</name>
        <value>false</value>
    </property>
</configuration>
```

hadoop2/etc/hadoop/ 디렉토리에서 vim 편집기
yarn-site.xml 파일 편집

[hadoop@localhost hadoop]\$ vim yarn-site.xml

<configuration></configuration> 부분 안에 수정해야함

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle,spark_shuffle</value>
    </property>
    <property>
        <name>yarn.nodemanager.aux-services.mapreduce_shuffle.class</name>
        <value>org.apache.hadoop.mapred.ShuffleHandler</value>
    </property>
    <property>
        <name>yarn.nodemanager.aux-services.spark_shuffle.class</name>
        <value>org.apache.spark.network.yarn.YarnShuffleService</value>
    </property>
    <property>
        <name>spark.yarn.shuffle.stopOnFailure</name>
        <value>false</value>
    </property>
    <property>
        <name>yarn.nodemanager.local-dirs</name>
        <value>/home/hadoop/yarn/nm-local-dir</value>
    </property>
    <property>
        <name>yarn.resourcemanager.fs.state-store.url</name>
        <value>/home/hadoop/data/yarn/system/rmstore</value>
    </property>
    <property>
        <name>yarn.resourcemanager.hostname</name>
        <value>jmj</value>
    </property>
    <property>
        <name>yarn.web-proxy.address</name>
        <value>0.0.0.0:8089</value>
    </property>
    <property>
        <name>yarn.nodemanager.recovery.enabled</name>
        <value>true</value>
    </property>
    <property>
        <name>yarn.nodemanager.address</name>
        <value>0.0.0.0:45454</value>
    </property>
    <property>
        <name>yarn.nodemanager.pmem-check-enabled</name>
        <value>false</value>
    </property>
    <property>
        <name>yarn.nodemanager.vmem-check-enabled</name>
        <value>false</value>
    </property>
</configuration>
```

완전 분산 모드

```
drwxrwxr-x. 2 hadoop hadoop 4096 11월  7 21:59 Downloads
drwxr-xr-x. 3 hadoop hadoop 4096 11월  7 23:31 data
drwxr-xr-x. 9 hadoop hadoop 4096 11월 14 2014 hadoop-2.6.0
-rw-rw-r--. 1 hadoop hadoop 195257604 11월  7 21:57 hadoop-2.6.0.tar.gz
lrwxrwxrwx. 1 hadoop hadoop      12 11월  9 22:16 hadoop2 -> hadoop-2.6.0
drwxr-xr-x. 3 hadoop hadoop 4096 11월  7 23:32 yarn
[hadoop@localhost ~]$ pwd
/home/hadoop
[hadoop@localhost ~]$ tar cvzf hadoop.tar.gz hadoop-2.6.0
hadoop-2.6.0/
hadoop-2.6.0/bin/
[hadoop@localhost ~]$ scp hadoop.tar.gz hadoop@192.168.100.110:/home/hadoop
hadoop.tar.gz
[hadoop@localhost ~]$ ll
합계 382992
```

```
drwxrwxr-x. 2 hadoop hadoop 4096 11월  7 21:59 Downloads
drwxr-xr-x. 3 hadoop hadoop 4096 11월  7 23:31 data
drwxr-xr-x. 9 hadoop hadoop 4096 11월 14 2014 hadoop-2.6.0
-rw-rw-r--. 1 hadoop hadoop 195257604 11월  7 21:57 hadoop-2.6.0.tar.gz
-rw-rw-r--. 1 hadoop hadoop 196904543 11월 10 00:15 hadoop.tar.gz
lrwxrwxrwx. 1 hadoop hadoop      12 11월  9 22:16 hadoop2 -> hadoop-2.6.0
drwxr-xr-x. 3 hadoop hadoop 4096 11월  7 23:32 yarn
[hadoop@localhost ~]$ exit
```

```
[hadoop@slave1 ~]$ tar xvzf hadoop.tar.gz
hadoop-2.6.0/
hadoop-2.6.0/bin/
```

설정이 완료된 hadoop을 압축하여 배포.

```
$tar cvzf Hadoop.tar.gz Hadoop-2.6.0
```

```
$scp Hadoop.tar.gz Hadoop@ip주소(또는 hostname):/home/Hadoop
```

배포받은 hadoop 압축 파일을 풀고 설치.

```
$ tar xvzf Hadoop.tar.gz
```

완전 분산 모드

1. zookeeper가 설치된 모든 서버에서 다음 과정 진행

```
[zookeeper@jmj ~]$ source .bashrc  
[zookeeper@jmj ~]$ zkServer.sh .start  
[zookeeper@jmj ~]$ zkServer.sh .status  
    Mode gkrdls : leader(1), follower(2)
```

2. jmj 호스트에서 zookeeper format

```
[hadoop@jmj ~]$ hdfs zkfc -formatZK
```

3. 각각의 journalnode 실행

```
[hadoop@jmj ~]$ hadoop-daemon.sh start journalnode
```

4. jmj 호스트에서 namenode 포맷

```
[hadoop@jmj ~]$ hdfs namenode -format
```

5. jmj 호스트에서 namenode 및 zkfc 실행

```
[hadoop@jmj ~]$ hadoop-daemon.sh start namenode
```

```
[hadoop@jmj ~]$ hadoop-daemon.sh start zkfc
```

6. jmj 호스트에서 datanodes 실행

```
[hadoop@jmj ~]$ hadoop-daemons.sh start datanode
```

완전 분산 모드

7. sdh 호스트에서 standbynamenode 및 zkfc 실행

```
[hadoop@sdh ~]$ hdfs namenode -bootstrapStandby  
[hadoop@sdh ~]$ hadoop-daemon.sh start namenode  
[hadoop@sdh ~]$ hadoop-daemon.sh start zkfc
```

8. jmj 호스트에서 yarn-cluster 실행

```
[hadoop@jmj ~]$ start-yarn.sh
```

9. jmj 호스트에서 historyserver 실행

```
[hadoop@jmj ~]$ mr-jobhistory-daemon.sh start historyserver
```

10. jmj 호스트에서 proxyserver 실행

```
[hadoop@jmj ~]$ yarn-demon.sh start proxyserver
```

11. 하둡 분산 정상 설치 여부 확인 (wordcount 실행)

```
[hadoop@jmj ~]$ hdfs haadmin -getServiceState nn1  
[hadoop@jmj ~]$ hdfs haadmin -getServiceState nn2  
[hadoop@jmj ~]$ hdfs dfs -put /home/hadoop/hadoop2/etc/hadoop/hadoop-env.sh /user/hadoop/conf  
[hadoop@jmj ~]$ yarn jaf\  
    /home/hadoop/hadoop2/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar\  
    wordcount conf output  
[hadoop@jmj ~]$ jps
```



Hadoop Eco System

(Spark | Zeppelin | Hive | Sqoop)

The Hive logo is the word "HIVE" in a bold, orange sans-serif font, with a stylized orange graphic of three overlapping rounded rectangles underneath it.

Hadoop Eco System

1. Spark 버전 확인

아래 사이트에 접속하여 설치하고자 하는 spark의 버전을 확인합니다.

```
http://spark.apache.org/downloads.html
```

2. Spark 2.1.3 설치

설치파일 다운로드

```
[hadoop@jmj ~]$ wget http://www.eu.apache.org/dist/spark/spark-2.1.3/spark-2.1.3-bin-hadoop2.6.tgz
```

압축 해제

```
[hadoop@jmj ~]$ tar -xvzf spark-2.1.3-bin-hadoop2.6.tgz
```

디렉토리명 변경

```
[hadoop@jmj ~]$ ln -s spark-2.1.3-bin-hadoop2.6 spark2
```

환경 변수 설정

```
[hadoop@jmj ~]$ vi .bashrc
```

```
export YARN_CONF_DIR=/home/hadoop/etc/hadoop  
export SPARK_HOME=/home/hadoop/spark2  
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin:$PATH
```

Hadoop Eco System

3. Spark on YARN

샘플 예제 실행

* cluster 모드는 다른 node에서 driver를 실행시키는 방식이고,
client 모드는 자신의 node에서 driver를 실행시키는 방식

client 방식 시행

```
[hadoop@jmj ~]$ spark-submit --deploy-mode cluster \
--class org.apache.spark.examples.SparkPi \
/home/hadoop/spark2/examples/jars/spark-examples*.jar
10
```

cluster 방식 시행

```
[hadoop@jmj ~]$ spark-submit --deploy-mode client \
--class org.apache.spark.examples.SparkPi \
/home/hadoop/spark2/examples/jars/spark-examples*.jar
10
```

참고

Spark in Standalone

Spark는 독립적인 모드로 설치 및 수행 가능

4. slaves에 설치/배포

```
[hadoop@jmj ~]$ tar cvzf spark.tar.gz spark-2.1.3-bin-hadoop2.6
[hadoop@jmj ~]$ scp spark.tar.gz hadoop@sdh:/home/hadoop
[hadoop@jmj ~]$ scp spark.tar.gz hadoop@khk:/home/hadoop
[hadoop@jmj ~]$ scp spark.tar.gz hadoop@lkh:/home/hadoop
```

5. slaves에서 압축 해제 및 설치

```
[hadoop@sdh ~]$ tar xvzf spark.tar.gz
[hadoop@sdh ~]$ ln -s spark-2.1.3-bin-hadoop2.6 spark2
[hadoop@khk ~]$ tar xvzf spark.tar.gz
[hadoop@khk ~]$ ln -s spark-2.1.3-bin-hadoop2.6 spark2
[hadoop@lkh ~]$ tar xvzf spark.tar.gz
[hadoop@lkh ~]$ ln -s spark-2.1.3-bin-hadoop2.6 spark2
```

이후 과정은 master 서버에서만 진행

Hadoop Eco System

6. slave 서버 등록

```
[hadoop@jmj ~]$ vim spark2/conf/slaves
```

```
sdh  
khk  
lkh
```

7. 환경 설정

```
[hadoop@jmj ~]$ vim spark2/conf/spark-site.xml
```

```
export JAVA_HOME=/usr/java/jdl1.7  
export HADOOP_CONF_DIR=/home/hadoop/hadoop2/etc/hadoop
```

```
[hadoop@jmj ~]$ vim spark2/conf/spark-de
```

```
spark.master          yarn  
spark.eventLog.enabled true  
spark.eventLog.dir    hdfs://jmj:9000/spark-logs  
spark.history.fs.logDirectory hdfs://jmj:9000/spark-logs  
spark.history.provider org.apache.spark.deploy.history.FsHistoryProvider  
spark.history.fs.update.interval 10  
spark.history.ui.port 18080
```

8. 서비스 시작/종료

```
[hadoop@jmj ~]$ start-master.sh  
[hadoop@jmj ~]$ start-slaves.sh
```

9. web ui for spark

```
http://192.168.103.143:8080
```

10. 테스트

```
[hadoop@jmj ~]$ spark-submit \  
--master spark://jmj:7077 \  
--class org.apache.spark.examples.SparkPi \  
/home/hadoop/spark2/examples/jars/spark-examples*.jar  
100
```

11. Spark history server

```
[hadoop@jmj ~]$ hdfs dfs -mkdir /spark-logs  
[hadoop@jmj ~]$ hdfs dfs -chmod /spark-logs  
[hadoop@jmj ~]$ hdfs dfs -chown hadoop:hadoop /spark-logs  
[hadoop@jmj ~]$ start-history-server.sh
```

12. web ui for spark history server:18080

Hadoop Eco System

The screenshot shows a web browser window with the following details:

- Address bar: 주의 요함 | mirror.apache-kr.org/zeppelin/zeppelin-0.8.0/
- Toolbar icons: App, Data_Science, G, SW, AWS, PA, Chart, JDK_API.
- Title: Index of /zeppelin/zeppelin-0.8.0
- Table of contents:

| Name | Last modified | Size | Description |
|--|------------------|------|-------------|
| Parent Directory | | - | |
| zeppelin-0.8.0-bin-all.tgz | 2018-06-24 19:22 | 938M | |
| zeppelin-0.8.0-bin-netinst.tgz | 2018-06-24 14:34 | 306M | |
| zeppelin-0.8.0.tgz | 2018-06-24 11:09 | 58M | |

1. Zeppelin 파일 다운 및 압축 해제

```
[hadoop@jmj ~]wget http://mirror.apache-kr.org/zeppelin/zeppelin-0.8.0/zeppelin-0.8.0-bin-all.tgz
[hadoop@jmj ~]tar zxfv zeppelin-0.8.0-bin-all.tgz
[hadoop@jmj ~]ln -s zeppelin-0.8.0-bin-all.tgz zeppelin
```

Hadoop Eco System

2. Zeppelin 설정

```
[hadoop@jmj ~]$ vi .bashrc
-----
export ZEPPELIN_HOME=/home/hadoop/zeppelin
export PATH=$PATH:$ZEPPELIN_HOME/bin

[zeppelin-env.sh]
[hadoop@jmj conf]$ cp zeppelin-env.sh.template zeppelin-env.sh
-----
export SPARK_HOME=/home/hadoop/zeppelin
export HADOOP_CONF_DIR=/home/hadoop/etc/hadoop2
export ZEPPELIN_PORT=8888
export JAVA_HOME=/usr/java/jdk1.7
export MASTER=spark://jmj:7077

[zeppelin-site.xml]
[hadoop@jmj conf]$ cp zeppelin-site.xml.template zeppelin-site.xml
-----
#Spark Web UI와 포트가 겹치니 수정해준다.

<property>
  <name>zeppelin.server.port</name>
  <value>8888</value>
</property>
```

```
[shiro.ini]
[hadoop@jmj conf]$ cp shiro.ini.template shiro.ini
```

#사용자 정보를 수정한다.

```
[users]
```

```
admin = admin, admin
```

```
[interpreter.json]
```

```
[hadoop@jmj interpreter]$ vim interpreter.json
#Spark 클러스터 정보를 입력한다.
#spark.Utils.InvokeMethod 장애 발생 시 useHiveContext 설정 값을 false로 변경한다.

"master": "spark://jmj:7077"
"zeppelin.spark.useHiveContext": "false"
```

3. zeppelin 실행

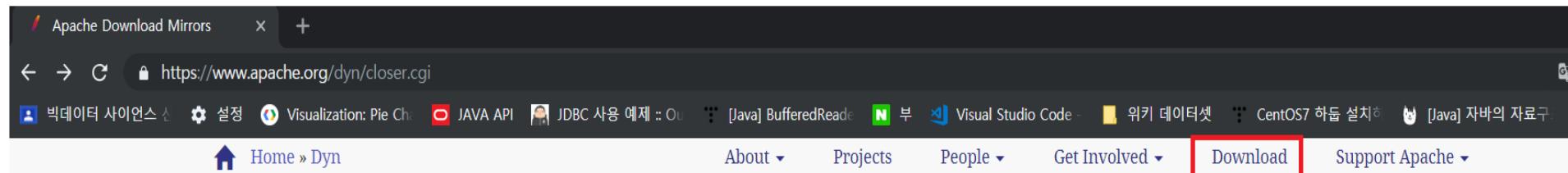
```
[hadoop@jmj interpreter]$zeppelin-daemon.sh start
```

4. web ui for zeppelin

```
http://192.168.103.143:8888
```

Hadoop Eco System

<http://www.apache.org/dyn/closer.cgi/hive/> 에 접속해서 [Download] – [http://apache.mirror.cdnetworks.com/]에 접속한다.



The screenshot shows a browser window with the URL <https://www.apache.org/dyn/closer.cgi>. The page is for the Apache Software Foundation. The navigation bar includes links for Home, About, Projects, People, Get Involved, Download (which is highlighted with a red box), and Support Apache. Below the navigation bar, there's a large Apache feather logo and the text "THE APACHE SOFTWARE FOUNDATION". To the right, there's a "SUPPORT APACHE" logo and a sidebar with links for Google Custom Search, The Apache Way, Contribute, and ASF Sponsors.



We suggest the following mirror site for your download:

<http://apache.mirror.cdnetworks.com/>

Other mirror sites are suggested below.

It is essential that you [verify the integrity](#) of the downloaded file using the PGP signature (`.asc` file) or a hash (`.md5` or `.sha*` file).

Please only use the backup mirrors to download KEYS, PGP and MD5 sigs/_hashes or if no other mirrors are working.

HTTP

<http://apache.mirror.cdnetworks.com/>

Hadoop Eco System

아파치 mirror 사이트에 들어가서 archive site를 클릭하면 아래와 같이 목록이 출력된다.
목록 중에 hive를 입력하고 들어간다.

Index of /dist

← → ⌂ 주의 요함 | archive.apache.org/dist/

비데이터 사이언스 설정 Visualization: Pie Ch JAVA API JDBC 사용 예제 :: Ou [Java] BufferedReade N 부 Visual Studio Code - 위키 데이터셋 CentOS7 하둡 설치하기 [Java] 자바의 자

Apache Software Foundation Distribution Directory

The directories linked below contain current software releases from the Apache Software Foundation projects. Older non-recommended releases can be found on our [archive site](#).

To find the right download for a particular project, you should start at the project's own webpage or on our [project resource listing](#) rather than browsing the links below.

Please do not download from apache.org! If you are currently at apache.org and would like to browse, please visit [a nearby mirror site](#) instead.

Projects

| Name | Last modified | Size | Description |
|----------------------------------|------------------|------|-------------|
| Parent Directory | - | - | |
| META/ | 2018-11-10 10:21 | - | |
| abdera/ | 2017-10-04 10:56 | - | |
| accumulo/ | 2018-10-16 19:22 | - | |
| ace/ | 2017-10-04 11:11 | - | |
| activemq/ | 2018-11-19 13:58 | - | |
| airavata/ | 2018-05-04 20:46 | - | |
| allura/ | 2018-10-30 14:19 | - | |
| ambari/ | 2018-11-16 17:46 | - | |
| ant/ | 2018-07-28 15:17 | - | |

Index of /dist

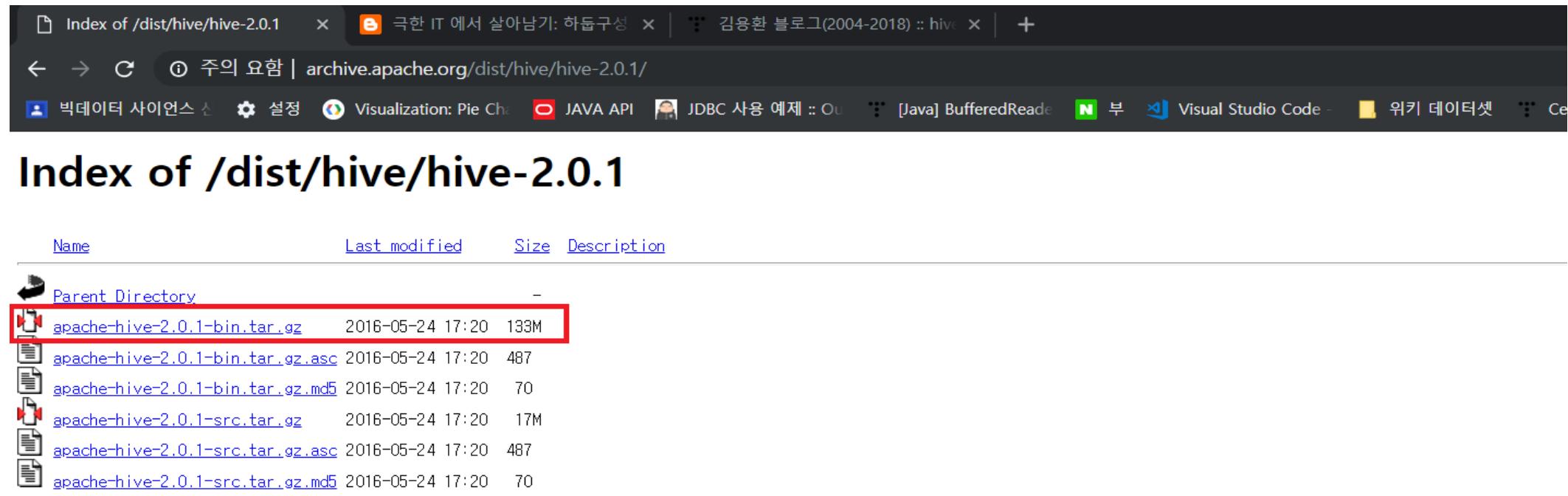
← → ⌂ 주의 요함 | archive.apache.org/dist/

비데이터 사이언스 설정 Visualization: Pie Ch JAVA API JDBC 사용 예제 :: Ou [Java] BufferedReade N 부 Visual Studio Code - 위키 데이터셋 CentOS7 하둡 설치하기 [Java] 자바의 자

| | | |
|---------------------------------|------------------|---|
| fLink/ | 2018-10-29 08:01 | - |
| flume/ | 2018-05-04 18:32 | - |
| fluo/ | 2018-05-04 19:35 | - |
| forrest/ | 2018-05-04 16:35 | - |
| freemarker/ | 2018-05-04 21:07 | - |
| geode/ | 2018-10-03 20:43 | - |
| geronimo/ | 2018-07-29 13:59 | - |
| giraph/ | 2018-05-04 17:33 | - |
| gora/ | 2018-05-04 16:41 | - |
| groovy/ | 2018-11-11 23:16 | - |
| guacamole/ | 2018-05-04 17:03 | - |
| hadoop/ | 2018-09-27 16:29 | - |
| hama/ | 2018-05-04 18:29 | - |
| harmony/ | 2017-10-04 10:45 | - |
| hawt/ | 2018-09-23 01:36 | - |
| hbase/ | 2018-11-20 00:39 | - |
| helix/ | 2018-07-30 18:13 | - |
| hive/ | 2018-11-06 06:50 | - |
| hivemind/ | 2017-10-04 11:10 | - |
| httpcomponents/ | 2018-05-04 18:50 | - |
| httled/ | 2018-10-26 18:53 | - |
| ibatis/ | 2017-10-04 10:47 | - |

Hadoop Eco System

Hive 폴더에 들어가서 hive-2.0.1 버전을 다운로드 받는다.



The screenshot shows a web browser window with the following details:

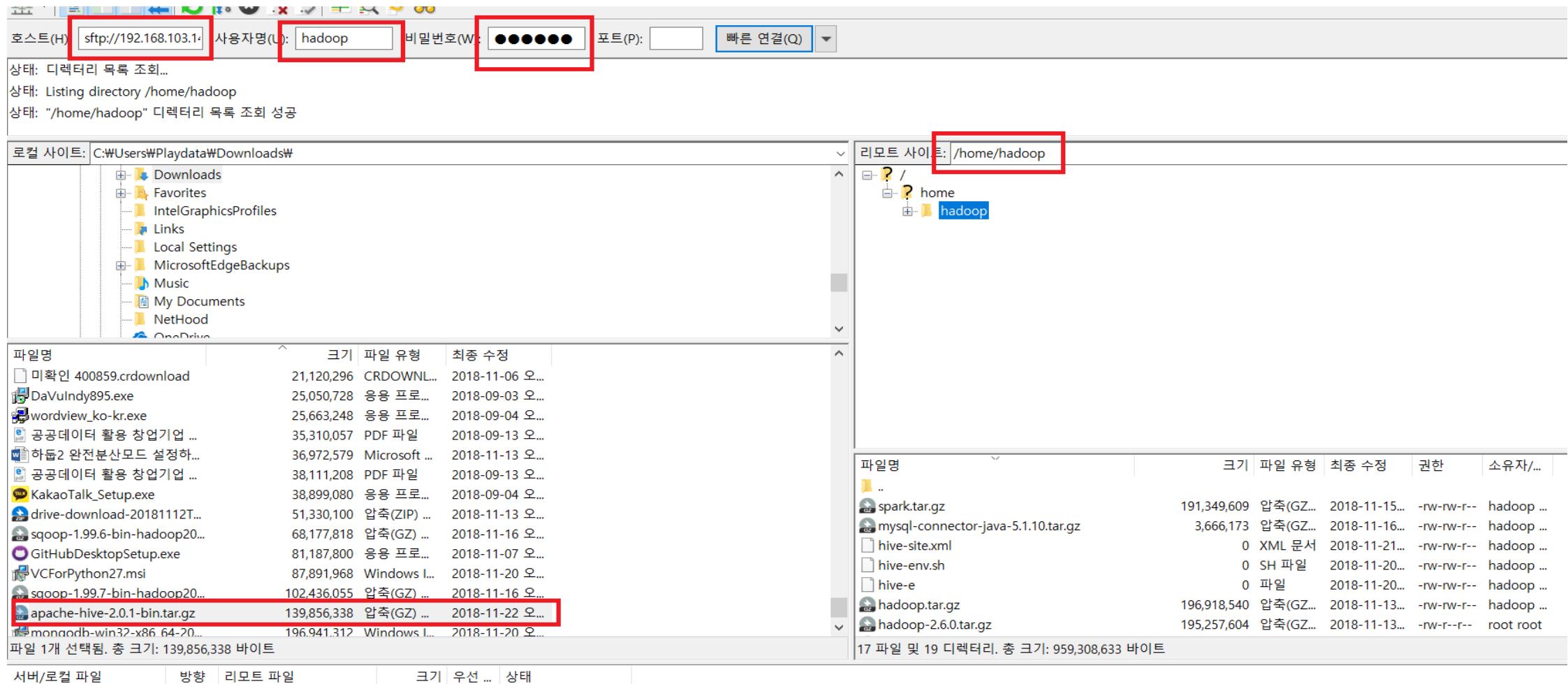
- Address bar: Index of /dist/hive/hive-2.0.1
- Address bar: 극한 IT에서 살아남기: 하둡구성
- Address bar: 김용환 블로그(2004-2018) :: hive
- Address bar: archive.apache.org/dist/hive/hive-2.0.1/
- Toolbar buttons: 빅데이터 사이언스, 설정, Visualization: Pie Chart, JAVA API, JDBC 사용 예제, [Java] BufferedReader, 부, Visual Studio Code, 위키 데이터셋, Ce

Index of /dist/hive/hive-2.0.1

| Name | Last modified | Size | Description |
|--|------------------|------|-------------|
| Parent Directory | | - | |
| apache-hive-2.0.1-bin.tar.gz | 2016-05-24 17:20 | 133M | |
| apache-hive-2.0.1-bin.tar.gz.asc | 2016-05-24 17:20 | 487 | |
| apache-hive-2.0.1-bin.tar.gz.md5 | 2016-05-24 17:20 | 70 | |
| apache-hive-2.0.1-src.tar.gz | 2016-05-24 17:20 | 17M | |
| apache-hive-2.0.1-src.tar.gz.asc | 2016-05-24 17:20 | 487 | |
| apache-hive-2.0.1-src.tar.gz.md5 | 2016-05-24 17:20 | 70 | |

Hadoop Eco System

Filezilla로 hadoop 아이디로 접속해서 /home/hadoop 폴더로 hive 파일을 전송한다.
Wget도 가능하다.



Hadoop Eco System

apache-hive-2.0.1-bin.tar.gz

Tar xvzf apache-hive-2.0.1-bin.tar.gz로 압축푼다.



```
2. 192.168.103.142
[hadoop@sdh ~]$ ls
${system:java.io.tmpdir}    data      downloads      hadoop.tar.gz  hive-env.sh  mysql-connector-java-5.1.10  spark2
apache-hive-2.0.1-bin.tar.gz  db-derby-10.4.2.0-bin  hadoop-2.6.0  hadoop2  hive-site.xml  mysql-connector-java-5.1.10.tar.gz  yarn
apache-hive-2.3.4-bin.tar.gz  derby      hadoop-2.6.0.tar.gz  hive-e  maven  spark.tar.gz
[hadoop@sdh ~]$ tar xvzf apache-hive-2.0.1-bin.tar.gz
```

Hadoop Eco System

apache-hive-2.0.1-bin.tar.gz을 압축 푼 후
\$ ln -s apache-hive-2.0.1-bin hive 로
해당 폴더를 hive 폴더로 축약한다.

```
[hadoop@sdh ~]$  
[hadoop@sdh ~]$ ls  
${system:java.io.tmpdir}    apache-hive-2.3.4-bin.tar.gz  derby      hadoop-2.6.0.tar.gz  hive-e      maven          spark.tar.gz  
apache-hive-2.0.1-bin       data                   downloads   hadoop.tar.gz     hive-env.sh  mysql-connector-java-5.1.10  spark2  
apache-hive-2.0.1-bin.tar.gz db-derby-10.4.2.0-bin  hadoop-2.6.0  hadoop2        hive-site.xml  mysql-connector-java-5.1.10.tar.gz  yarn  
[hadoop@sdh ~]$ ln -s apache-hive-2.0.1-bin hive  
[hadoop@sdh ~]$ ls  
${system:java.io.tmpdir}    data           hadoop-2.6.0      hive          maven          spark2  
apache-hive-2.0.1-bin      db-derby-10.4.2.0-bin  hadoop-2.6.0.tar.gz  hive-e      mysql-connector-java-5.1.10  yarn  
apache-hive-2.0.1-bin.tar.gz derby          hadoop.tar.gz    hive-env.sh  mysql-connector-java-5.1.10.tar.gz  
apache-hive-2.3.4-bin.tar.gz downloads      hadoop2        hive-site.xml  spark.tar.gz  
[hadoop@sdh ~]$
```

Hadoop Eco System

```
$ cd ~  
$ vi .bashrc로 들어가서  
export HIVE_HOME=/home/hadoop/hive  
export HIVE_CONF_DIR=$HIVE_HOME/conf  
export PATH=$PATH:$HIVE_HOME/bin  
를 입력한 후 wq로 저장하고 나온다.  
그리고 $ source .bashrc 를 입력해서 환경설정 파일을 적용해준다.
```

```
# export SYSTEMD_PAGER=  
  
# User specific aliases and functions  
  
export JAVA_HOME=/usr/java/jdk1.7  
#export PATH=$JAVA_HOME/bin:$PATH  
export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:/home/hadoop/hadoop2/sbin:/home/hadoop/spark2/bin:/home/hadoop/spark2/sbin:$PATH  
  
[red box]  
| export HIVE_HOME=/home/hadoop/hive  
| export PATH=$PATH:$HIVE_HOME/bin:$PATH  
[red box]  
| export HIVE_CONF_DIR=$HIVE_HOME/conf  
| export HIVE_CLASS_PATH=$HIVE_CONF_DIR  
| export HADOOP_USER_CLASSPATH_FIRST=true  
| export HADOOP_HOME=/home/hadoop/hadoop2  
| export CLASSPATH=$CLASSPATH:/home/hadoop/hadoop2/lib/*:  
| export CASSSPATH=$CLASSPATH:/usr/local/hive/lib/*:  
  
[red box]  
| export DERBY_HOME=/home/hadoop/derby  
| export DERBY_INSTALL=/home/hadoop/derby  
| export PATH=$PATH:$DERBY_HOME/bin  
  
| export CLASSPATH=$CLASSPATH:$DERBY_HOME/lib/derby.jar:$DERBY_HOME/lib/derbytools.jar  
| export SPARK_HOME=/home/hadoop/spark2  
  
|  
| export ECLIPSE_HOME=/usr/java/eclipse  
| export PATH=$PATH:$ECLIPSE_HOME  
| export CLASSPATH=$JAVA_HOME/lib/tools.jar  
| export JAVA_HOME CLASSPATH  
| export SPARK_SUBMIT=/home/hadoop/spark2/bin/spark-submit  
| export PATH
```

Hadoop Eco System

```
#MySQL 연결 라이브러리 추가  
$ wget http://repo.maven.apache.org/maven2/mysql/mysql-connector-java/5.1.9/mysql-connector-java-5.1.9.jar  
  
#MySQL 연결 라이브러리 복사  
$ mv mysql-connector-java-5.1.9.jar ./lib  
  
#hive-site.xml 파일 수정  
$ vi conf/hive-site.xml  
  
<?xml version="1.0"?><?xml-stylesheet type="text/xsl" href="configuration.xsl"?><configuration>  
<property>  
    <name>hive.metastore.local</name>  
    <value>true</value>  
</property>  
<property>  
    <name>javax.jdo.option.ConnectionURL</name>  
    <value>jdbc:mysql://localhost:3306/metastore?createDatabaseIfNotExist=true&useUnicode=true&characterEncoding=UTF-8</value>  
</property>  
<property>  
    <name>javax.jdo.option.ConnectionDriverName</name>  
    <value>com.mysql.jdbc.Driver</value>  
</property>  
<property>  
    <name>javax.jdo.option.ConnectionUserName</name>  
    <value>username</value>  
</property>  
<property>  
    <name>javax.jdo.option.ConnectionPassword</name>  
    <value>password</value>  
</property>  
<property>  
    <name>hive.metastore.uris</name>  
    <value>thrift://localhost:10000</value>  
</property>  
</configuration>
```

Hadoop Eco System

```
$ cp conf/hive-log4j.properties.template conf/hive-log4j.properties  
$ vi conf/hive-log4j.properties
```

아래 변수를 찾아 다르면 아래 항목으로 수정

```
log4j.appender.EventCounter=org.apache.hadoop.log.metrics.EventCounter
```

하둡 디렉토리 구성: hive.metastore.warehouse.dir 설정(./conf/hive-default.xml) 가능.

```
$ hadoop fs -mkdir /tmp  
$ hadoop fs -mkdir /user/hive/warehouse  
$ hadoop fs -chmod g+w /tmp  
$ hadoop fs -chmod g+w /user/hive/warehouse3.
```

Hive 실행

서버로 실행

```
$ hive --service hiveserver &  
$ hive --service metastore &
```

실행

```
$ hive
```

|

Hadoop Eco System

```
1 -- sqoop을 설치할 디렉토리 생성  
2  
3 [root@host명 home]# mkdir sqoop  
4     sqoop 설치할 디렉토리 생성  
5  
6 [root@host명 home]# cd sqoop  
7     sqoop 디렉토리로 이동
```

```
1 -- sqoop을 설치 및 변경  
2  
3 http://apache.mirror.cdnetworks.com/sqoop/1.4.7/sqoop-1.4.7.bin_hadoop-2.6.0.tar.gz  
4     apache.mirror에서 sqoop-1.4.7.bin_hadoop-2.6.0.tar.gz를 다운로드 받은 후 filezilla를 통해 파일 옮기기  
5  
6 [root@host명 sqoop]# tar xvfz sqoop-1.4.7.bin_hadoop-2.6.0.tar.gz  
7     파일 압축 해제  
8  
9 [root@host명 sqoop]# ln -s sqoop-1.4.7.bin_hadoop-2.6.0 sqoop2  
10    파일 이름 변경
```

Hadoop Eco System

```
1 -- sqoop 경로 설정
2
3 [root@host명 sqoop]# vim /home/hadoopo/.bashrc
4     export PATH=$JAVA_HOME/bin:/home/hadoop/hadoop2/bin:/home/hadoop/hadoop2/sbin:
5             /home/hadoop/spark2/bin:/home/hadoop/spark2/sbin:/home/sqoop/sqoop/bin:$PATH
6     export SQOOP_HOME=/home/sqoop/sqoop
7
8     --> sqoop 설치경로를 hadoop에 경로 설정
9
10
11 [root@host명 ~]# source /home/hadoop/.bashrc
12     수정된 사항 적용
```

```
1 -- sqoop 경로 설정
2
3 [root@host명 sqoop2]# vim sqoop-env-template.sh
4     #Set path to where bin/hadoop is available
5     export HADOOP_COMMON_HOME=/home/hadoop/hadoop2
6
7     #Set path to where hadoop-* core.jar is available
8     export HADOOP_MAPRED_HOME=/home/hadoop/hadoop2
9
10    --> hadoop 경로 설정
11
12 [root@host명 sqoop2]# source sqoop-env-template.sh
13     수정된 사항 적용
```

Hadoop Eco System

```
1 -- sqoop 실행 오류
2
3 [root@host명 ~]# sqoop list-tables --connect jdbc:mysql://localhost:3306/test
4   --username 'mysql계정명' --password 'mysql계정의 비밀번호'
5
6 아래 오류의 해결 방법은 /home/sqoop/sqoop/lib에 mysql 드라이버를 설치하면 해결됨
7   ERROR sqoop.Sqoop: Got exception running Sqoop: java.lang.RuntimeException:
8     Could not load db driver class: com.mysql.jdbc.Driver
```

```
1 -- sqoop 실행 오류 해결법
2
3 https://dev.mysql.com/downloads/file/?id=481158
4   mysql 디렉토리를 다운로드하고 filezilla를 이용해 lib 디렉토리 위치에 설치
5
6 [root@host명 ~]# cd /home/sqoop/sqoop2/lib
7   디렉토리
8
9 [root@host명 lib]# tar xvfz mysql-connector-nodejs-8.0.13.tar.gz
10  압축 해제
11
12 [root@host명 ~]# vim /home/hadoop/.bashrc
13   CLASSPATH=.:${JAVA_HOME}/lib/tools.jar:${JAVA_HOME}/home/sqoop/sqoop2/lib/package
14   export CLASSPATH
15   경로 설정
16
17 [root@host명 ~]# source /home/hadoop/.bashrc
18   수정된 사항 적용
```

완전 분산 모드로 진행
(Use Ambari)



완전 분산 모드

본 장은 하둡 관리 소프트웨어인 Ambari를 이용한 내용을 기술합니다.

CentOS 7을 설치 및 네트워크 설정 과정은 생략하고 난 후의
설치 및 설정 과정을 기술합니다.

완전 분산 모드

Host명 변경

```
1 [root@localhost ~]# hostnamectl set-hostname [변경할 host명]  
2 [root@localhost ~]# reboot
```

Fully Qualified Domain Name 설정

```
1 [root@host명 ~]# vi /etc/hosts
```

```
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4  
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6  
  
# FQDN  
192.168.103.151 namenode.hadoop.com namenode  
192.168.103.152 datanode01.hadoop.com datanode01  
192.168.103.153 datanode02.hadoop.com datanode02  
192.168.103.154 datanode03.hadoop.com datanode03  
192.168.103.155 datanode04.hadoop.com datanode04
```

완전 분산 모드

방화벽 설정 및 해제

```
1 [root@host명 ~]# yum install firewalld  
2 [root@host명 ~]# systemctl stop firewalld  
3 [root@host명 ~]# systemctl disable firewalld
```

SSH Key 생성 및 SSH 접속 설정

```
1 [root@host명 ~]# ssh-keygen  
2 [root@host명 ~]# cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

```
1 [root@host명 ~]# cat ~/.ssh/id_rsa.pub | ssh root@namenode.hadoop.com 'cat >> ~/.ssh/authorized_keys'  
2 [root@host명 ~]# cat ~/.ssh/id_rsa.pub | ssh root@datanode01.hadoop.com 'cat >> ~/.ssh/authorized_keys'  
3 [root@host명 ~]# cat ~/.ssh/id_rsa.pub | ssh root@datanode02.hadoop.com 'cat >> ~/.ssh/authorized_keys'  
4 [root@host명 ~]# cat ~/.ssh/id_rsa.pub | ssh root@datanode03.hadoop.com 'cat >> ~/.ssh/authorized_keys'  
5 [root@host명 ~]# cat ~/.ssh/id_rsa.pub | ssh root@datanode04.hadoop.com 'cat >> ~/.ssh/authorized_keys'
```

완전 분산 모드

Hugepage 편집

```
1 [root@host명 ~]# vi /etc/rc.local
```

```
1 if test -f /sys/kernel/mm/transparent_hugepage/enabled; then  
2 /bin/echo never > /sys/kernel/mm/transparent_hugepage/enabled  
3 fi  
4 if test -f /sys/kernel/mm/transparent_hugepage/defrag; then  
5 /bin/echo never > /sys/kernel/mm/transparent_hugepage/defrag  
6 fi
```

SELinux 비활성화 설정

```
1 [root@host명 ~]# vi /etc/selinux/config
```

```
1 SELINUX=enforcing --> SELINUX=disabled
```

완전 분산 모드

Swap 활용도 설정

```
1 [root@host명 ~]# vi /etc/sysctl.conf
```

```
1 vm.swappiness=10 <-- 추가
```

Ambari repository 다운로드

```
1 wget -nv http://public-repo-1.hortonworks.com/ambari/centos7/
2 .x/updates/2.6.1.0/ambari.repo -o /etc/yum.repos.d/ambari.repo
```

Ambari server 다운로드

```
1 yum install ambari-server
```

완전 분산 모드

MySQL 연동

```
1 [root@host명 ~]# yum install mysql-connector-java  
2 [root@host명 ~]# ambari-server setup --jdbc-db=mysql --jdbc-dirver=/usr/share/java/mysql-connector-java.jar
```

Ambari server 설정

```
1 [root@host명 ~]# ambari-server setup
```

Ambari agent 다운로드

```
1 [root@host명 ~]# yum install ambari-agent
```

완전 분산 모드

ambari agent 설정

```
1 [root@host명 ~]# vi /etc/ambari-agent/ambari-agent.ini
```

```
1 hostname=localhost --> hostname=master의 host명  
2  
3 [security] 부분에 아래 내용 추가  
4 force_https_protocol=PROTOCOL_TLSv1_2
```

```
1 [root@host명 ~]# vi /etc/python/cert-verification.cfg
```

```
1 [https]  
2 verify=platform_default --> verify=disable
```

완전 분산 모드

Host 간 NTP 시간 동기화

```
1 [root@host명 ~]# yum remove chrony  
2 [root@host명 ~]# yum install ntp  
3 [root@host명 ~]# systemctl stop ntpd
```

NTP 설정

```
1 [root@host명 ~]# vi /etc/ntp.conf
```

```
1 server 0.centos.pool.ntp.org iburst  
2 server 1.centos.pool.ntp.org iburst  
3 server 2.centos.pool.ntp.org iburst  
4 server 3.centos.pool.ntp.org iburst  
5  
6 주석 후 아래 내용 추가  
7  
8 server 'namenode의 ip주소' iburst
```

완전 분산 모드

Windows hosts 파일 설정

```
1 | 메모장을 관리자 권한으로 실행한 후 C:\Windows\System32\drivers\etc\에  
2 | 위치한 hosts 파일의 내용 수정  
3 |  
4 | ambari-server에 할당한 ip와 host명 등록  
5 | ex) xxx.xxx.xxx.xxx master.hadoop.com
```

Ambari server 실행

```
1 | [root@host명 ~]# ambari-server start
```

Ambari agent 실행

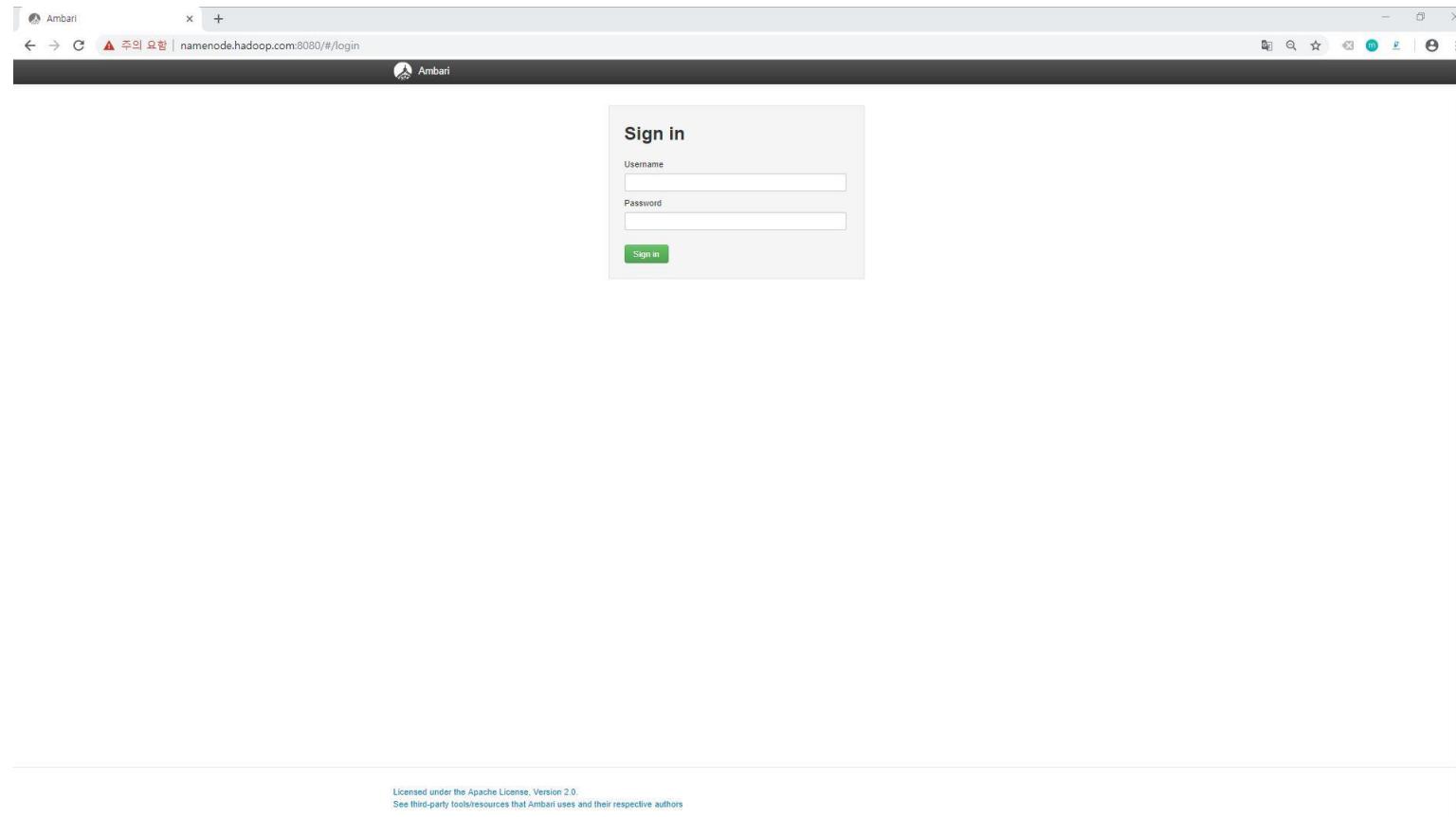
```
1 | [root@host명 ~]# ambari-agent start
```

완전 분산 모드

Web browser에서 Ambari 실행 및 접속

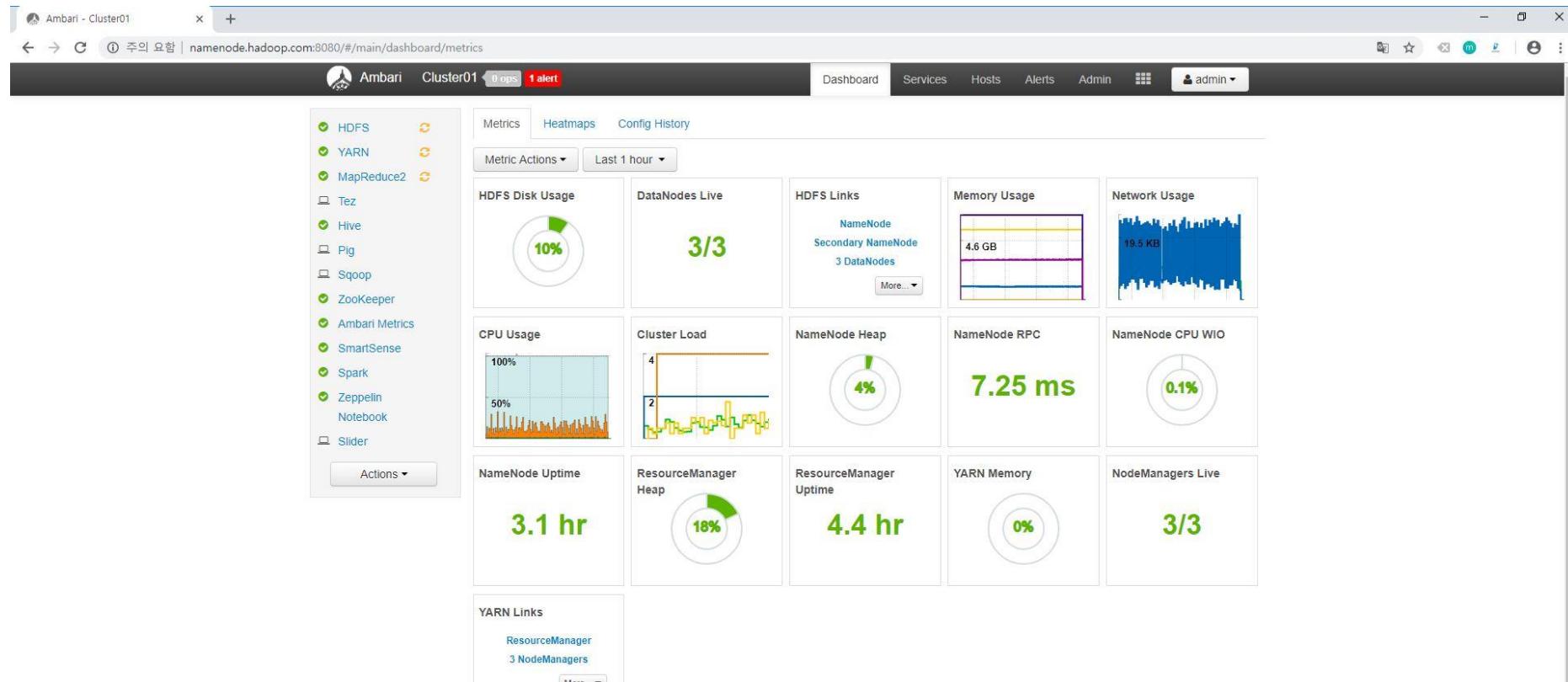
1 | http://'master의 host명 또는 master의 ip':8080

1 | username : admin, password : admin 입력 후 login



완전 분산 모드

Ambri 메인 화면



완전 분산 모드

Cluster에 등록된 Hosts 목록 화면

| Name | IP Address | Rack | Cores | RAM | Disk Usage | Load Avg | Versions | Components |
|-----------------------|-----------------|--------------|-------|--------|------------|-------------|---------------|------------|
| datanode01.hadoop.com | 192.168.103.152 | default-rack | 2 (2) | 7.64GB | 0.11 | HDP-2.6.5.0 | 11 Components | |
| datanode02.hadoop.com | 192.168.103.153 | default-rack | 2 (2) | 5.67GB | 0.40 | HDP-2.6.5.0 | 16 Components | |
| datanode03.hadoop.com | 192.168.103.154 | default-rack | 2 (2) | 5.67GB | 1.23 | HDP-2.6.5.0 | 18 Components | |
| datanode04.hadoop.com | 192.168.103.155 | default-rack | 2 (2) | 5.67GB | 0.28 | HDP-2.6.5.0 | 17 Components | |
| namenode.hadoop.com | 192.168.103.151 | default-rack | 2 (2) | 7.64GB | 1.79 | HDP-2.6.5.0 | 20 Components | |

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

mysql-connector-java 권한 변경 (namenode host에서 진행)

```
1 [root@host명 ~]# chmod 644 /usr/share/java/mysql-connector-java.jar
```

mysql repo 다운로드 (hive 서비스를 설치할 node에서 진행)

```
1 [root@Host명 ~]# wget http://repo.mysql.com/mysql-community-release-el7-5.noarch.rpm  
2 [root@Host명 ~]# sudo rpm -ivh mysql-community-release-el7-5.noarch.rpm
```

mysql server 다운로드 (hive 서비스를 설치할 node에서 진행)

```
1 [root@Host명 ~]# yum install mysql-server
```

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

mysql 시작 (hive 서비스를 설치할 node에서 진행)

```
1 | [root@Host명 ~]# sudo systemctl start mysqld
```

mysql 사용 (hive 서비스를 설치할 node에서 진행)

```
1 | [root@Host명 ~]# mysql -u root -p  
2 |      root 계정의 처음 비밀번호는 없으므로 비밀번호는 입력하는 부분이 나왔을 때 그냥 'enter key' 입력
```

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

root 계정으로 mysql을 접속한 후 아래 내용 작성 후 exit로 종료 (새로운 계정 생성 및 사용을 위한 설정)

```
1 mysql> CREATE USER 'mysql에서 사용할 계정명'@'%' IDENTIFIED BY 'mysql에서 사용할 계정명의 비밀번호';
2 mysql> GRANT ALL PRIVILEGES ON *.* TO 'mysql에서 사용할 계정명'@'%';
3 mysql> CREATE USER 'mysql에서 사용할 계정명'@'localhost' IDENTIFIED BY 'mysql에서 사용할 계정명의 비밀번호';
4 mysql> GRANT ALL PRIVILEGES ON *.* TO 'mysql에서 사용할 계정명'@'localhost';
5 mysql> CREATE USER 'mysql에서 사용할 계정명'@'hive 서비스를 설치할 node host명' IDENTIFIED BY 'mysql에서 사용할 계정명의 비밀번호';
6 mysql> GRANT ALL PRIVILEGES ON *.* TO 'mysql에서 사용할 계정명'@'hive 서비스를 설치할 node의 host명';
7 mysql> FLUSH PRIVILEGES;
8 mysql> exit;
```

```
1 ex)
2 mysql> CREATE USER 'ambari'@'%' IDENTIFIED BY 'ambari';
3 mysql> GRANT ALL PRIVILEGES ON *.* TO 'ambari'@'%';
4 mysql> CREATE USER 'ambari'@'localhost' IDENTIFIED BY 'ambari';
5 mysql> GRANT ALL PRIVILEGES ON *.* TO 'ambari'@'localhost';
6 mysql> CREATE USER 'ambari'@'datanode03.hadoop.com' IDENTIFIED BY 'ambari';
7 mysql> GRANT ALL PRIVILEGES ON *.* TO 'ambari'@'datanode03.hadoop.com';
8 mysql> FLUSH PRIVILEGES;
9 mysql> exit;
```

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

새로 생성한 계정으로 mysql 접속

```
1 [root@host ~]# mysql -u 새로운 계정명 -p  
2     ex) mysql -u ambari -p
```

다음 mysql 설정 진행

```
1 mysql> CREATE DATABASE 새로 생성할 DB명;  
2 mysql> USE 새로 생성한 DB명;  
3 mysql> SOURCE Ambari-DDL-MySQL-CREATE.sql;
```

```
1 ex)  
2 mysql> CREATE DATABASE ambari;  
3 mysql> USE ambari;  
4 mysql> SOURCE Ambari-DDL-MySQL-CREATE.sql;  
5 별도로 첨부한 Ambari-DDL-MySQL-CREATE.sql 파일을 편집기로 실행한 뒤  
6 전체 내용을 복사 붙여넣기 한 후 종료
```

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

Ambari 관리 Page에서 서비스 추가

Oozie Server

Oozie Server host stampede03a

Oozie Database

- New Derby Database
- Existing MySQL Database
- Existing PostgreSQL Database
- Existing Oracle Database

Be sure you have run:
ambari-server setup --jdbc-db=postgres --jdbc-driver=/path/to/postgres/driver.jar on the Ambari Server host to make the JDBC driver available and to enable testing the database connection.

Database Host stampede01a.com

Database Name oozie

Database Username oozie

Database Password *****

JDBC Driver Class org.postgresql.Driver

Database URL jdbc:postgresql://stampede01a.com:5432/oozie

Test Connection Connection OK

1 Oozie Database
2 사용할 DB 선택
3 MySQL을 미리 설치했을 경우, Existing MySQL Database 선택
4
5 [Database Host] : Hive를 설치할 host
6 [Database Name] : 사용할 db 명
7 [Database Username] : 사용할 mysql의 사용자 명
8 [Database Password] : 사용할 mysql의 사용자 password
9
10 위 사항을 입력한 Test Connection 실행 --> Connection OK가 뜨면 정상 실행
11 화면 하단에 위치한 next 버튼 실행

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

Ambari 관리 Page에서 Hive View를 실행하기 위한 설정 (namenode host에서 진행)

```
1 [root@host명 ~]# su hdfs
2 [hdfs@host명 root]$ hdfs dfs -chown admin:hadoop /user/${username}
3 ${username} : ambari 관리 페이지에 접속해 사용하는 사용자 명
4     ex) hdfs dfs -chown admin:hadoop /user/admin
5
6 reference url
7 https://docs.hortonworks.com/HDPDocuments/Ambari-2.6.2.0/bk\_ambari-views/content/setup\_HDFS\_user\_directory.html
```

Hive 실행

```
1 [root@host명 ~]# su hdfs
2 [hdfs@host명 root]$ hive
3     hive 실행 명령 동작 시 시간 다소 소요
4
5 hive>
```

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

Ambari 관리자 계정인 admin의 정보가 저장되어 있는 위치 확인

```
1 [root@host명 ~]# cd /lib/ambari-server/web/data/users  
2  
3 [root@host명 users]# ls  
4     privileges.json  privileges_admin.json  user_admin.json  user_user.json  users.json  
5  
6 [root@host명 users]# cat privileges_admin.json
```

완전 분산 모드 (Host에 MySQL 및 Hive 설치)

Ambari 관리자 계정인 admin의 정보가 저장되어 있는 위치 확인

```
hdfs@namenode:/lib/ambari-server/web/data/users
[root@namenode users]# cat privileges_admin.json
{
  "href" : "http://c6401.ambari.apache.org:8080/api/v1/privileges?PrivilegeInfo/principal_name=admin&fields=PrivilegeInfo/*",
  "items" : [
    {
      "href" : "http://c6401.ambari.apache.org:8080/api/v1/privileges/1",
      "PrivilegeInfo" : {
        "permission_name" : "AMBARI.ADMINISTRATOR",
        "principal_name" : "admin",
        "principal_type" : "USER",
        "privilege_id" : 1
      }
    }
  ]
}
[root@namenode users]#
```



**Thank you very much
for your attention**