■ 하둡2 - 기본 환경 설정

○ bigdata 계정으로 Server01에 접속해 하둡2 다운로드

```
[bigdata@server01 ~]$
wget http://mirror.navercorp.com/apache/hadoop/common/stable2/hadoop-2.9.1.tar.gz
```

○ 하둡2 압축풀기

```
[bigdata@server01 ~]$ tar xvfz hadoop-2.9.1.tar.gz
```

○ 하둡2 환경설정 파일 수정을 위해 hadoop-2.9.1/etc/hadoop으로 이동

- etc/hadoop 안에 환경설정 파일들이 모두 들어있음

```
[bigdata@server01 ~]$ cd hadoop-2.9.1/etc/hadoop/
```

```
[bigdata@server01 ~]$ cd hadoop2/etc/hadoop/
[bigdata@server01 hadoop]$ ls
capacity-scheduler.xml      httpfs-env.sh            mapred-env.sh
configuration.xsl           httpfs-log4j.properties  mapred-queues.xml.template
container-executor.cfg      httpfs-signature.secret  mapred-site.xml.template
core-site.xml               httpfs-site.xml          slaves
hadoop-env.cmd              kms-acls.xml             ssl-client.xml.example
hadoop-env.sh               kms-env.sh               ssl-server.xml.example
hadoop-metrics.properties   kms-log4j.properties     yarn-env.cmd
hadoop-metrics2.properties  kms-site.xml             yarn-env.sh
hadoop-policy.xml           log4j.properties         yarn-site.xml
hdfs-site.xml               mapred-env.cmd
```

○ vi 에디터로 hadoop-env.sh 수정 후 저장(:wq)

- 하둡을 실행하는 셸 스크립트 파일에서 필요한 환경변수 설정

```
[bigdata@server01 hadoop]$ vi hadoop-env.sh
```

```
# 실제 JDK가 설치된 경로로 수정
export JAVA_HOME=/usr/local/java

#하둡 데몬의 PID 저장 경로를 다음과 같이 수정
export HADOOP_PID_DIR=/home/bigdata/hadoop-2.9.1/pids
```

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

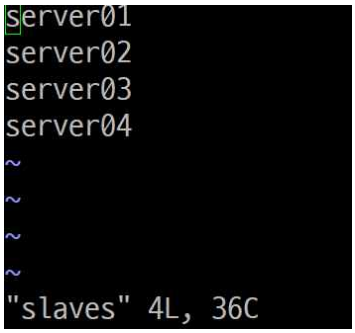
```
export HADOOP_PID_DIR=/home/bigdata/hadoop-2.9.1/pids
export HADOOP_SECURE_DN_PID_DIR=${HADOOP_PID_DIR}
```

```
[bigdata@server01 hadoop]$ source hadoop-env.sh
```

○ vi 에디터로 slaves 파일에 데이터 노드 호스트 목록 설정 후 저장(:wq)

```
[bigdata@server01 hadoop]$ vi slaves
```

```
server01
server02
server03
server04
```



○ vi 에디터로 core-site.xml 작성 후 저장 (:wq)

- <configuration></configuration> 사이에 작성
- HDFS와 MapReduce에서 공통적으로 사용할 환경 정보 설정

```
[bigdata@server01 hadoop]$ vi core-site.xml
```

```
<!--HDFS 전체 이름 value를 통해 datanode가 namenode를 알수 있음-->
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://servers-cluster</value>
  </property>

<!--네임노드HA를 구성할 때 사용할 주키퍼 서버의 주소 -->
  <property>
    <name>ha.zookeeper.quorum</name>
    <value>server01:2181,server02:2181,server03:2181</value>
  </property>
```

○ vi 에디터를 이용해 hdfs-site.xml 수정후 저장(:wq)

- <configuration></configuration> 사이에 작성

- HDFS 환경정보 수정

```
[bigdata@server01 hadoop]$ vi hdfs-site.xml
```

```
  <property>
    <name>dfs.replication</name>
    <value>3</value>
  </property>

  <property>
    <name>dfs.blocksize</name>
    <value>67108864</value>
  </property>

  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/home/bigdata/data/dfs/namenode</value>
  </property>

  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/home/bigdata/data/dfs/datanode</value>
  </property>

  <property>
    <name>dfs.journalnode.edits.dir</name>
    <value>/home/bigdata/data/dfs/journalnode</value>
  </property>

<property>
    <name>dfs.nameservices</name>
    <value>servers-cluster</value>
  </property>

  <property>
    <name>dfs.ha.namenodes.servers-cluster</name>
    <value>nn1,nn2</value>
  </property>

  <property>
    <name>dfs.namenode.rpc-address.servers-cluster.nn1</name>
    <value>server01:8020</value>
  </property>
 <property>
    <name>dfs.namenode.rpc-address.servers-cluster.nn2</name>
    <value>server02:8020</value>
  </property>
```

```
  <property>
    <name>dfs.namenode.http-address.servers-cluster.nn1</name>
    <value>server01:50070</value>
  </property>

<property>
    <name>dfs.namenode.http-address.servers-cluster.nn2</name>
    <value>server02:50070</value>
  </property>

  <property>
    <name>dfs.namenode.shared.edits.dir</name>
    <value>qjournal://server01:8485;server02:8485;server03:8485/servers-cluster</value>
  </property>

  <property>
    <name>dfs.client.failover.proxy.provider.servers-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/bigdata/.ssh/id_rsa</value>
  </property>

  <property>
    <name>dfs.ha.automatic-failover.enabled</name>
    <value>true</value>
  </property>
```

○ mapred-site.xml를 수정해야하는 데 템플릿 파일만 있기때문에 mapred-site.xml 파일을 만듦

```
[bigdata@server01 hadoop]$ cp mapred-site.xml.template mapred-site.xml
```

○ vi 에디터로 mapred-site.xml를 수정한뒤 저장(:wq)

- <configuration></configuration> 사이에 작성

- 맵리듀스 환경정보 수정

```
[bigdata@server01 hadoop]$ vi mapred-site.xml
```

```
<!--맵리듀스 잡을 어떤 모드로 실행할지 설정함. local모드와 yarn모드가 있음  -->
<property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
```

○ vi 에디터로 yarn-site.xml를 수정한뒤 저장(:wq)

- <configuration></configuration> 사이에 작성

- yarn 환경 정보 수정

```
[bigdata@server01 hadoop]$ vi yarn-site.xml
```

```
<!-- 노드매니저 간의 서비스 제어 -->
 <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>

<!-- yarn.nodemanager.aux-services 서비스를 구현한 클래스 설정 -->
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce_shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>

<!--노드매니저가 애플리케이션을 실행할 때 필요한 파일을 저장하는 로컬 파일 시스템 경로
-->
  <property>
    <name>yarn.nodemanager.local-dirs</name>
    <value>/home/bigdata/data/yarn/nm-local-dir</value>
  </property>

<!--리소스매니저의 상태 정보를 저장할 로컬 파일 시스템 경로 -->
  <property>
    <name>yarn.resourcemanager.fs.state-store.uri</name>
    <value>/home/bigdata/data/yarn/system/rmstore</value>
  </property>

<!--리소스매니저의 호스트명 설정 -->
<property>
    <name>yarn.resourcemanager.hostname</name>
```

```
    <value>server01</value>
  </property>

<!--애플리케이션마스터에 대한 사용자 접근을 제어하기 위한 웹프록시 서버 -->
  <property>
    <name>yarn.web-proxy.address</name>
    <value>0.0.0.0.8089</value>
  </property>
```

○ 하둡 배포하기 위해 이제까지 설정한 하둡 파일을 Tar파일로 압축

```
[bigdata@server01 ~]$ cd ~
[bigdata@server01 ~]$ tar cvfz hadoop.tar.gz hadoop-2.9.1/
```

○ 압축 파일을 다른 서버에 모두 배포

```
[bigdata@server01 ~]$ scp hadoop.tar.gz bigdata@server02:/home/bigdata
scp hadoop.tar.gz bigdata@server03:/home/bigdata
scp hadoop.tar.gz bigdata@server04:/home/bigdata
ssh bigdata@server02 "cd /home/bigdata; tar xvfz hadoop.tar.gz"
ssh bigdata@server03 "cd /home/bigdata; tar xvfz hadoop.tar.gz"
ssh bigdata@server04 "cd /home/bigdata; tar xvfz hadoop.tar.gz"
```

○ 주키퍼 실행전 방화벽 확인

```
# systemctl status firewalld.service : firewalld 작동 상태 확인
# systemctl stop firewalld :  firewalld 중지(방화벽)
# systemctl disable firewalld : firewalld 자동 시작 중지
```

○ 하둡 실행 전 bigdata 계정으로 Server01, Server02, Server03에 있는 주키퍼 실행

```
[bigdata@server01 ~]$ ./zookeeper-3.4.12/bin/zkServer.sh start
```

```
[bigdata@server02 ~]$ ./zookeeper-3.4.12/bin/zkServer.sh start
```

```
[bigdata@server03 ~]$ ./zookeeper-3.4.12/bin/zkServer.sh start
```

○ ZKFC를 실행하기 전 주키퍼 초기화

```
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs zkfc -formatZK
```

```
18/05/30 11:54:20 INFO zookeeper.ZooKeeper: Session: 0x3000006e6730000 closed
18/05/30 11:54:20 WARN ha.ActiveStandbyElector: Ignoring stale result from old c
lient with sessionId 0x3000006e6730000
18/05/30 11:54:20 INFO zookeeper.ClientCnxn: EventThread shut down
18/05/30 11:54:20 INFO tools.DFSZKFailoverController: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down DFSZKFailoverController at server01/192.168.56.101
************************************************************/
```

■ 하둡2 실행

- hadoop 명령어는 하둡 홈디렉터리에 있는 bin 디렉터리안에 있음

bigdata 계정으로 server01, server02, server03 서버에 저널노드 실행

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start journalnode
```

```
[bigdata@server02 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start journalnode
```

```
[bigdata@server03 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start journalnode
```

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start journalnode
starting journalnode, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-bigdata-
journalnode-server01.out
```

만약 연결 거부 오류가 나면 etc/hadoop/hadoop-env.sh 환경 설정 파일을 적용

```
[bigdata@server01 hadoop-2.9.1]$ source etc/hadoop/hadoop-env.sh
```

만약 Error: Cannot find configuration directory: /etc/hadoop 오류 발생시,

root 계정으로 server01에 접속 후 /etc/profile 하단에 아래와 같이 작성

```
# vi /etc/profile
```

```
export PATH=$PATH:$HOME/hadoop-2.9.1/bin
export HADOOP_HOME=/home/bigdata/hadoop-2.9.1
```

```
# source /etc/profile
```

- sever01 에서 네임노드를 초기화 하고 모든 데몬을 실행하면 하둡 실행

```
[bigdata@server01 ~]$ cd hadoop-2.9.1
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs namenode -format
```

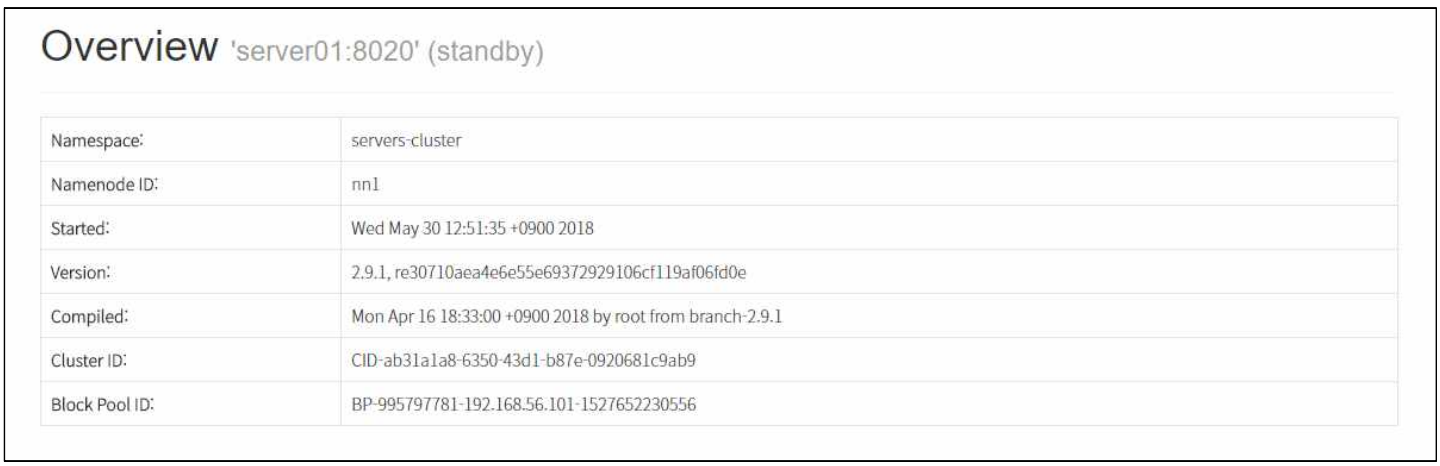
```
18/05/30 10:05:47 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = server01/192.168.56.101
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 2.9.1
```

액티브 네임노드 실행

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start namenode
```

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start namenode
starting namenode, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-bigdata-nam
enode-server01.out
```

- 웹브라우저에서 http://server01:50070에 접속해 namenode 확인

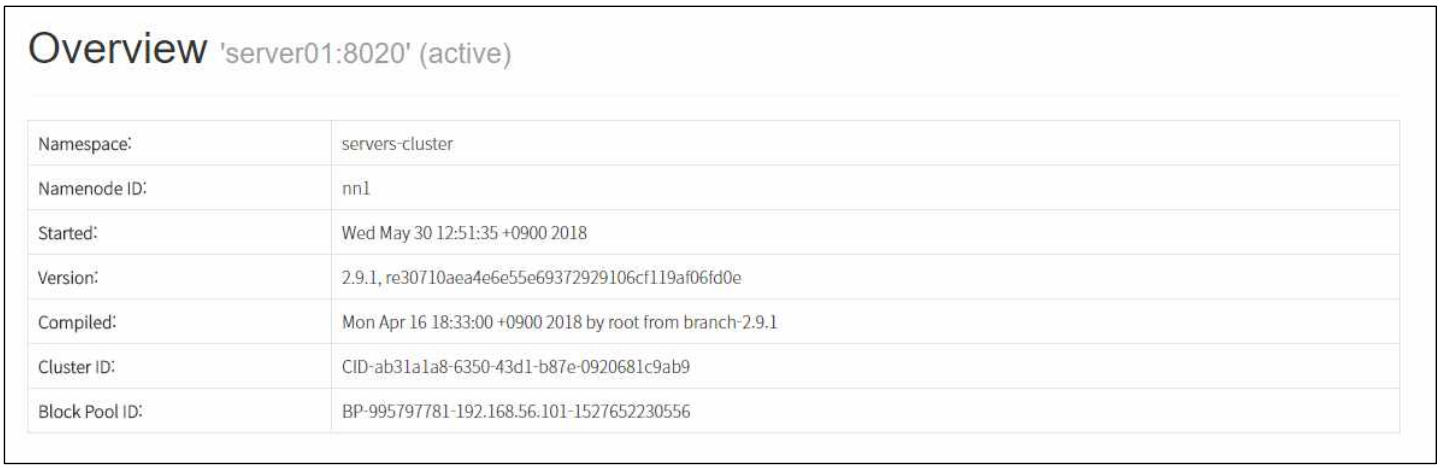


- 처음 네임노드를 실행하면 standby
- 주키퍼 장애컨트롤러(ZKFC)를 실행해야 active로 결정됨

ZKFC 실행

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start zkfc
```

- 네임노드 웹 인터페이스를 다시 확인해보면 active로 바뀌었음



전체 데이터 노드 실행

- hadoop-daemon 이 아닌 hadoop-daemons로 전체 데이터 노드 실행

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/hadoop-daemons.sh start datanode
```

```
server02: starting datanode, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-b
igdata-datanode-server02.out
server03: starting datanode, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-b
igdata-datanode-server03.out
server01: starting datanode, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-b
igdata-datanode-server01.out
server04: starting datanode, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-b
igdata-datanode-server04.out
```

스탠바이 네임노드 설정하기 위해 server02에 bigdata 계정으로 로그인 후 하둡 디렉터리로 이동

```
[bigdata@server02 ~]$ cd hadoop-2.9.1/
```

액티브 네임노드의 메타데이터를 복사

-bootstrapStandby 명령어: 스탠바이 네임노드 포맷 및 액티브 네임노드의 메타데이터가 스탠바이 네임노드로 복사됨

```
[bigdata@server02 hadoop-2.9.1]$ ./bin/hdfs namenode -bootstrapStandby
```

```
18/05/30 13:08:23 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = server02/192.168.56.102
STARTUP_MSG:   args = [-bootstrapStandby]
STARTUP_MSG:   version = 2.9.1
```

```
=====================================================
About to bootstrap Standby ID nn2 from:
           Nameservice ID: servers-cluster
        Other Namenode ID: nn1
  Other NN's HTTP address: http://server01:50070
  Other NN's IPC  address: server01/192.168.56.101:8020
             Namespace ID: 378103558
           Block pool ID: BP-995797781-192.168.56.101-1527652230556
              Cluster ID: CID-ab31a1a8-6350-43d1-b87e-0920681c9ab9
          Layout version: -63
       isUpgradeFinalized: true
=====================================================
```

스탠바이 네임노드 실행

```
[bigdata@server02 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start namenode
```

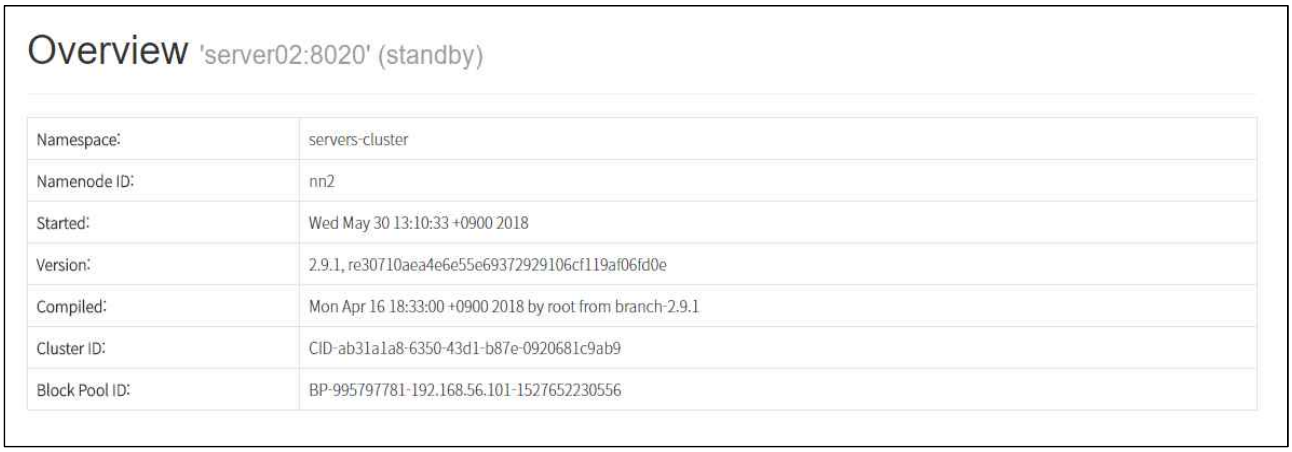
```
starting namenode, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-bigdata-namenode-ser
ver02.out
```

스탠바이용 ZKFC 실행

```
[bigdata@server02 hadoop-2.9.1]$ ./sbin/hadoop-daemon.sh start zkfc
```

```
starting zkfc, logging to /home/bigdata/hadoop-2.9.1/logs/hadoop-bigdata-zkfc-server02.ou
t
```

- 웹브라우저에서 http://server02:50070에 접속해 namenode 상태 확인



○ HDFS 정상적으로 작동하는 지 테스트

HDFS 홈디렉터리 만들기

- 홈디렉터리: /user/로그인한 계정

```
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs dfs -ls /
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs dfs -mkdir /user
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs dfs -mkdir /user/bigdata
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs dfs -mkdir /user/bigdata/conf
```

HDFS 디렉터리 확인

```
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs dfs -ls -R /
```

```
drwxr-xr-x   - bigdata supergroup          0 2018-05-30 13:15 /user
drwxr-xr-x   - bigdata supergroup          0 2018-05-30 13:15 /user/bigdata
drwxr-xr-x   - bigdata supergroup          0 2018-05-30 13:15 /user/bigdata/conf
```

하둡 환경설정 파일 업로드

```
[bigdata@server01 hadoop-2.9.1]$
 ./bin/hdfs dfs -put etc/hadoop/hadoop-env.sh /user/bigdata/conf/
```

업로드 파일 확인

```
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs dfs -ls conf
```

```
[bigdata@server01 hadoop-2.9.1]$ ./bin/hdfs dfs -ls conf
Found 1 items
-rw-r--r--   1 bigdata supergroup       5003 2018-05-25 11:27 conf/hadoop-env.sh
```

○ 얀 클러스터 실행

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/start-yarn.sh
```

```
starting yarn daemons
starting resourcemanager, logging to /home/bigdata/hadoop-2.9.1/logs/yarn-bigdata-reso
urcemanager-server01.out
server03: starting nodemanager, logging to /home/bigdata/hadoop-2.9.1/logs/yarn-bigdat
a-nodemanager-server03.out
server02: starting nodemanager, logging to /home/bigdata/hadoop-2.9.1/logs/yarn-bigdat
a-nodemanager-server02.out
server04: starting nodemanager, logging to /home/bigdata/hadoop-2.9.1/logs/yarn-bigdat
a-nodemanager-server04.out
server01: starting nodemanager, logging to /home/bigdata/hadoop-2.9.1/logs/yarn-bigdat
a-nodemanager-server01.out
```

○ 맵리듀스 잡 이력을 확인 할 수 있는 historyserver 시작

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/mr-jobhistory-daemon.sh start historyserver
```

○ 하둡 설치 확인

- jps: JVM 프로세스 상태를 확인하는 명령어

```
$ jps
```

```
[bigdata@server01 hadoop-2.9.1]$ jps
4914 NodeManager
4084 NameNode
5252 JobHistoryServer
4245 DFSZKFailoverController
4389 DataNode
4805 ResourceManager
5349 Jps
3943 JournalNode
1932 QuorumPeerMain
```

```
[bigdata@server02 hadoop-2.9.1]$ jps
1904 DataNode
1682 QuorumPeerMain
2531 Jps
2165 DFSZKFailoverController
1814 JournalNode
2042 NameNode
2299 NodeManager
```

```
[bigdata@server03 ~]$ jps
2113 Jps
1685 QuorumPeerMain
1879 DataNode
1994 NodeManager
1789 JournalNode
```

```
[bigdata@server04 ~]$ jps
1856 NodeManager
1997 Jps
1742 DataNode
```

○ 하둡 2 빌드

- 하둡은 일부 기능을 C로 구현함
- 이런 기능을 구현한 컴포넌트를 하둡 네이티브 라이브러리라고 함
- 하둡 설치 파일에는 32비트 리눅스 운영체제가 기본으로 지원
- 하둡을 전부 종료

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/stop-all.sh
```

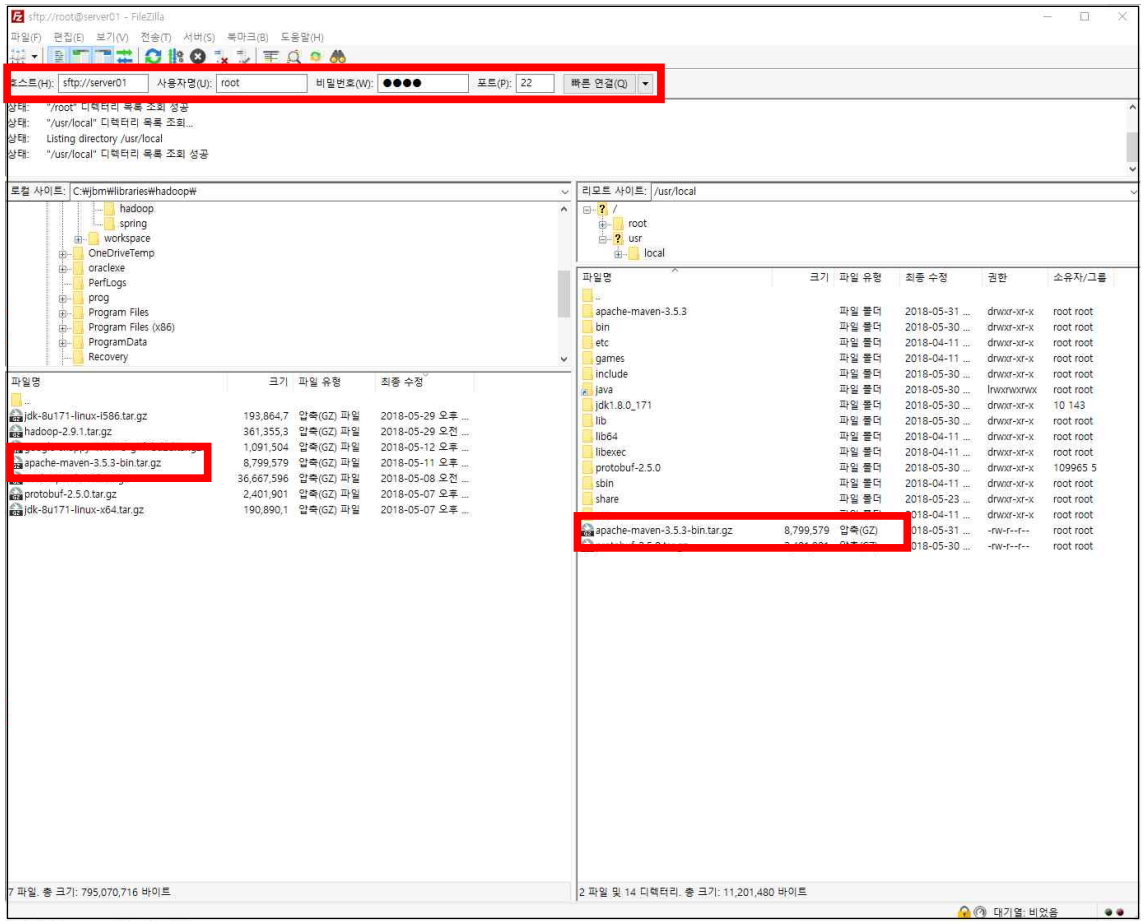
```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/mr-jobhistory-daemon.sh stop historyserver
```

○ server01에 bigdata로 접속해 하둡 소스 다운로드

```
$ wget http://mirror.navercorp.com/apache/hadoop/common/hadoop-2.9.1/hadoop-2.9.1-src.tar.gz
```

○ 메이븐 설치를 위해 파일질라로 /usr/local에 메이븐 tar 파일을 업로드

```
호스트 : sftp://server01
사용자명 : root
비밀번호 : 1111
포트 : 22
```



○ root계정으로 server01에 로그인한 뒤 /usr/local 디렉토리로 가서 메이븐 tar 파일 압축 풀기

```
[root@server01 local]# cd /usr/local
[root@server01 local]# tar xvfz apache-maven-3.5.3-bin.tar.gz
```

○ 압축이 풀린 메이븐 디렉토리를 리눅스 프로파일(/etc/profile)에 추가 후 적용

```
[root@server01 local]# vi /etc/profile
export PATH=$PATH:/usr/local/apache-maven-3.5.3/bin
[root@server01 local]# source /etc/profile
```

```
export PATH=$PATH:$JAVA_HOME/bin
export CLASS_PATH="."
export PATH=$PATH:/usr/local/apache-maven-3.5.3/bin
```

○ mvn 명령어를 실행했을 때, 메이븐 관련 로그가 출력되면 설치 된 것임

```
[root@server01 ~]# mvn

[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 0.262 s
[INFO] Finished at: 2018-05-25T14:36:14+09:00
[INFO] ------------------------------------------------------------------------
[ERROR] No goals have been specified for this build. You must specify a valid lifecycle phase or a goal in the format <plugin-prefix>:<goal> or <plugin-group-id>:<plugin-artifact-id>[:<plugin-version>]:<goal>. Available lifecycle phases are: validate, initialize, generate-sources, process-sources, generate-resources, process-resources, compile, process-classes, generate-test-sources, process-test-sources, generate-test-resources, process-test-resources, test-compile, process-test-classes, test, prepare-package, package, pre-integration-test, integration-test, post-integration-test, verify, install, deploy, pre-clean, clean, post-clean, pre-site, site, post-site, site-deploy. -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/NoGoalSpecifiedException
```

○ 메이븐 버전 확인

```
[root@server01 ~]# mvn --version
```

```
[root@server01 ~]# mvn --version
Apache Maven 3.5.3 (3383c37e1f9e9b3bc3df5050c29c8aff9f295297; 2018-02-25T04:49:05+09:00)
Maven home: /usr/local/apache-maven-3.5.3
Java version: 1.8.0_171, vendor: Oracle Corporation
Java home: /usr/local/jdk1.8.0_171/jre
Default locale: ko_KR, platform encoding: UTF-8
OS name: "linux", version: "3.10.0-693.el7.x86_64", arch: "amd64", family: "unix"
```

○ cmake 설치후 확인

- 리눅스에서 소스를 컴파일 할 경우 configure -> make -> make install 단계
- CMAKE는 configure 단계를 대체

```
[root@server01 ~]# yum -y install cmake
[root@server01 ~]# cmake --version
```

```
[root@server01 ~]# cmake --version
cmake version 2.8.12.2
```

○ zlib 설치 확인

```
[root@server01 ~]# ls -al /usr/lib64/libz.so.1
```

```
[root@server01 ~]# ls -al /usr/lib64/libz.so.1
lrwxrwxrwx. 1 root root 13  5월  7 16:13 /usr/lib64/libz.so.1 -> libz.so.1.2.7
```

○ openSSL 패키지 설치

- openSSL: 웹브라우저와 서버 간의 통신을 암호화하는 오픈소스 라이브러리

```
[root@server01 ~]# yum -y install openssl-devel
```

○ bigdata계정으로 server01 서버에 로그인 후 하둡 소스 파일 압축 풀기

```
[bigdata@server01 ~]$ tar xvfz hadoop-2.9.1-src.tar.gz
```

○ 하둡 소스 파일 디렉토리로 이동한 뒤 빌드 진행

```
[bigdata@server01 ~]$ cd hadoop-2.9.1-src
[bigdata@server01 ~]$ mvn package -Pdist,native -DskipTests -Dtar
```

○ "hadoop-2.9.1-src/hadoop-dist/target" 에서 생성된 "hadoop-2.9.1" 디렉토리와 "hadoop-2.9.1.tar.gz" 파일 확인

```
[bigdata@server01 ~]$ cd hadoop-2.9.1-src/hadoop-dist/target
```

○ "hadoop-2.9.1/lib/native" 네이티브 라이브러리에 생성된 네이티브 파일을 하둡이 설치된 디렉터리로 복사

```
$ cp hadoop-2.9.1/lib/native/* /home/bigdata/hadoop-2.9.1/lib/native
```

○ 하둡을 전부 실행

```
[bigdata@server01 hadoop-2.9.1]$ ./sbin/start-all.sh
[bigdata@server01 hadoop-2.9.1]$ ./sbin/mr-jobhistory-daemon.sh start historyserver
[bigdata@server01 hadoop-2.9.1]$ ./sbin/yarn-daemon.sh start proxyserver
```