

## ■ 하둡2 - 기본 환경 설정

- root 계정으로 4개의 서버에 로그인 한 뒤, 서버 인코딩 방식 확인
  - 하둡은 인코딩 방식으로 UTF-8을 사용
  - 인코딩 방식이 다르다면 문자열을 처리할 때 문제 발생

```
# echo $LANG
```

```
[root@server01 ~]# echo $LANG
ko_KR.UTF-8
```

- UTF-8이 아니라면 인코딩 변경

```
[root@server01 ~]# vi /etc/sysconfig/i18n
LANG="ko_KR.UTF-8"
SUPPORTED="en_US.UTF-8:en_US:ko_KR.encKR:ko_KR:ko"
SYSFONT="lat0-sun16"
SYSFONTACM="8859-15"
```

```
LANG="ko_KR.UTF-8"
SUPPORTED="en_US.UTF-8:en_US:ko_KR.encKR:ko_KR:ko"
SYSFONT="lat0-sun16"
SYSFONTACM="8859-15"
```

- source 명령어를 이용해 수정한 i18n 파일을 시스템에 적용

```
[root@server01 ~]# source /etc/sysconfig/i18n
```

- echo 명령어로 LANG 파라미터 확인

```
# echo $LANG
```

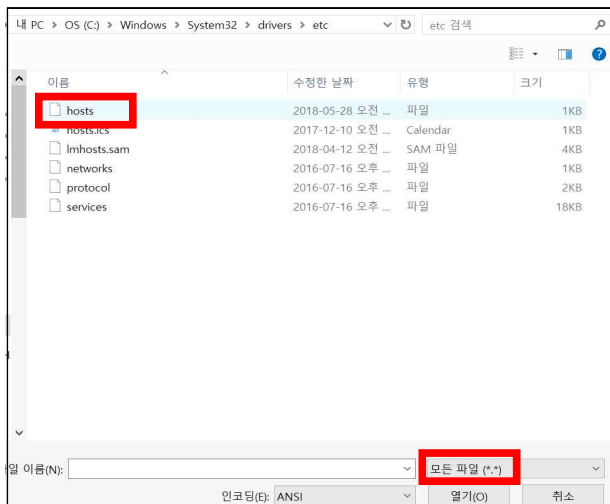
```
[root@server01 ~]# echo $LANG
ko_KR.UTF-8
```

2) 메모장을 마우스 오른쪽 버튼으로 클릭 후 ‘관리자 권한으로 실행’

3) ‘파일’ > ‘열기’ 클릭 후 C:\Windows\System32\drivers\etc 로 이동

C:\Windows\System32\drivers\etc

4) ‘모든 파일(\*.\*)’ 선택 후 ‘host’ 파일 열기



5) 'hosts'파일 하단에 아래처럼 입력

- IP와 호스트 이름을 매핑

```
192.168.56.101 server01
192.168.56.102 server02
192.168.56.103 server03
192.168.56.104 server04
```

## 6) 자바를 설치하기 위해 오라클로 이동해 리눅스용 JDK8를 다운받음

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Java SE Development Kit 8u171		
You must accept the <a href="#">Oracle Binary Code License Agreement for Java SE</a> to download this software.		
<input checked="" type="radio"/> Accept License Agreement <input type="radio"/> Decline License Agreement		
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.97 MB	<a href="#">jdk-8u171-linux-arm32-vfp-hflt.tar.gz</a>
Linux ARM 64 Hard Float ABI	74.89 MB	<a href="#">jdk-8u171-linux-arm64-vfp-hflt.tar.gz</a>
Linux x86	170.05 MB	<a href="#">jdk-8u171-linux-i586.rpm</a>
Linux x86	184.88 MB	<a href="#">jdk-8u171-linux-i586.tar.gz</a>
Linux x64	167.14 MB	<a href="#">jdk-8u171-linux-x64.rpm</a>
Linux x64	182.05 MB	<a href="#">jdk-8u171-linux-x64.tar.gz</a>
Mac OS X x64	247.84 MB	<a href="#">jdk-8u171-macosx-x64.dmg</a>
Solaris SPARC 64-bit (SVR4 package)	139.83 MB	<a href="#">jdk-8u171-solaris-sparcv9.tar.Z</a>
Solaris SPARC 64-bit	99.19 MB	<a href="#">jdk-8u171-solaris-sparcv9.tar.gz</a>
Solaris x64 (SVR4 package)	140.6 MB	<a href="#">jdk-8u171-solaris-x64.tar.Z</a>
Solaris x64	97.05 MB	<a href="#">jdk-8u171-solaris-x64.tar.gz</a>
Windows x86	199.1 MB	<a href="#">jdk-8u171-windows-i586.exe</a>
Windows x64	207.27 MB	<a href="#">jdk-8u171-windows-x64.exe</a>

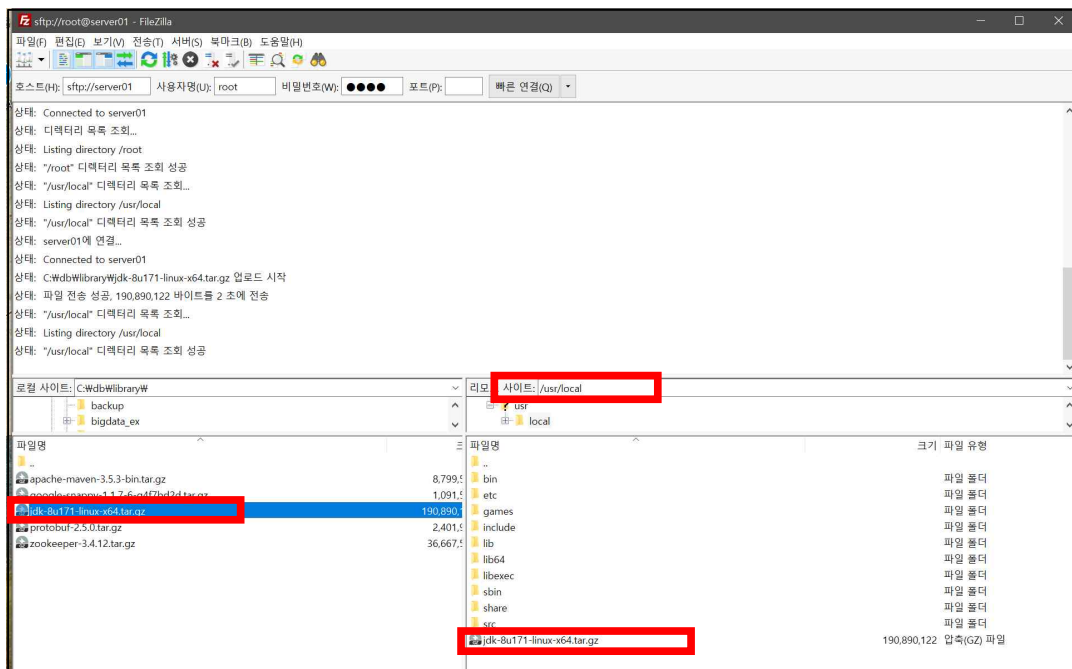
## 7) 파일질라를 이용해 JDK를 서버(경로 /usr/local)에 업로드

호스트 : server01

사용자명 : root

비밀번호: 1111

포트 : 22



## 8) Server01에 root 계정으로 로그인

## 9) 다운로드 디렉토리에 접근

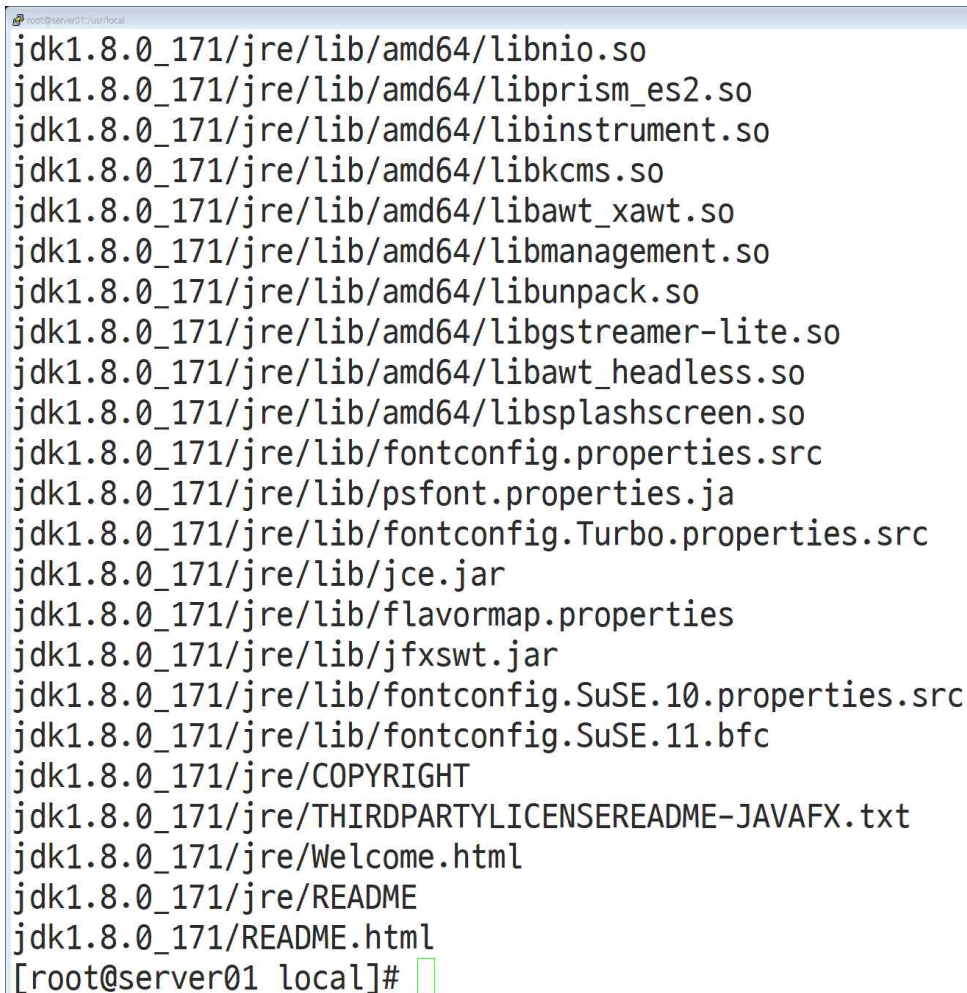
```
[root@server01 ~]# cd /usr/local/
```

- 권한문제일 경우 권한 변경

```
[root@server01 ~]# chmod 755 jdk-8u171-linux-x64.tar.gz
```

## 10) Tar 파일 압축 풀기

```
[root@server01 ~]# tar xvfz jdk-8u171-linux-x64.tar.gz
```



```
jdk1.8.0_171/jre/lib/amd64/libnio.so
jdk1.8.0_171/jre/lib/amd64/libprism_es2.so
jdk1.8.0_171/jre/lib/amd64/libinstrument.so
jdk1.8.0_171/jre/lib/amd64/libkcms.so
jdk1.8.0_171/jre/lib/amd64/libawt_xawt.so
jdk1.8.0_171/jre/lib/amd64/libmanagement.so
jdk1.8.0_171/jre/lib/amd64/libunpack.so
jdk1.8.0_171/jre/lib/amd64/libgstreamer-lite.so
jdk1.8.0_171/jre/lib/amd64/libawt_headless.so
jdk1.8.0_171/jre/lib/amd64/libsplashscreen.so
jdk1.8.0_171/jre/lib/fontconfig.properties.src
jdk1.8.0_171/jre/lib/psfont.properties.ja
jdk1.8.0_171/jre/lib/fontconfig.Turbo.properties.src
jdk1.8.0_171/jre/lib/jce.jar
jdk1.8.0_171/jre/lib/flavormap.properties
jdk1.8.0_171/jre/lib/jfxswt.jar
jdk1.8.0_171/jre/lib/fontconfig.SuSE.10.properties.src
jdk1.8.0_171/jre/lib/fontconfig.SuSE.11.bfc
jdk1.8.0_171/jre/COPYRIGHT
jdk1.8.0_171/jre/THIRDPARTYLICENSEREADME-JAVAFX.txt
jdk1.8.0_171/jre/Welcome.html
jdk1.8.0_171/jre/README
jdk1.8.0_171/README.html
[root@server01 local]#
```

## 11) JDK 경로를 빨리 찾을 수 있게 심볼릭 링크 생성

- 버전이 바뀔 경우 모든 설정파일에서 변경하는 것보다 심볼릭 링크만 수정

```
[root@server01 ~]# ln -s jdk1.8.0_171 java
```

## 12) ls 명령어로 심볼릭 링크 확인

```
[root@server01 ~]# ls -al
```

```
[root@server01 local]# ls -al
합계 186420
drwxr-xr-x. 13 root root      197  5월 23 16:07 .
drwxr-xr-x. 13 root root      155  5월  7 16:13 ..
drwxr-xr-x.  2 root root         6 11월  6 2016 bin
drwxr-xr-x.  2 root root         6 11월  6 2016 etc
drwxr-xr-x.  2 root root         6 11월  6 2016 games
drwxr-xr-x.  2 root root         6 11월  6 2016 include
lrwxrwxrwx.  1 root root         12  5월 23 16:07 java -> jdk1.8.0_171
-rwxr-xr-x.  1 root root 190890122  5월 23 15:51 jdk-8u171-linux-x64.tar.gz
drwxr-xr-x.  8  10  143        255  3월 29 09:18 jdk1.8.0_171
drwxr-xr-x.  2 root root         6 11월  6 2016 lib
drwxr-xr-x.  2 root root         6 11월  6 2016 lib64
drwxr-xr-x.  2 root root         6 11월  6 2016 libexec
drwxr-xr-x.  2 root root         6 11월  6 2016 sbin
drwxr-xr-x.  5 root root         49  5월  7 16:13 share
drwxr-xr-x.  2 root root         6 11월  6 2016 src
```

## 13) /etc/profile 파일을 vi 편집기로 열기

```
[root@server01 ~]# vi /etc/profile
```

## 14) 다음과 같이 /etc/profile 파일 가장 아래에 자바 경로 관련 환경 변수 등록

```
export JAVA_HOME=/usr/local/java
export PATH=$PATH:$JAVA_HOME/bin
export CLASS_PATH="."
```

```
unset -f pathmunge
```

```
export JAVA_HOME=/usr/local/java
export PATH=$PATH:$JAVA_HOME/bin
export CLASS_PATH="."
```

15) profile 파일 수정 후 source 명령어를 통해 변경된 profile을 시스템에 적용

```
[root@server01 ~]# source /etc/profile
```

16) 자바 버전 확인

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

```
[root@server01 ~]# java -version
java version "1.8.0_171"
Java(TM) SE Runtime Environment (build 1.8.0_171-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.171-b11, mixed mode)
```

17) 모든 서버에 JDK 설치

- 위의 과정을 나머지 3개의 서버에서도 똑같이 실행



18) **Server01**에서 **bigdata** 계정으로 로그인하고 서버가 다른 서버로 키 입력 없이 이동하기 위해 SSH 공개키를 생성

- 네임노드가 데이터노드에 자유롭게 접근하기 위해서

```
[bigdata@server01 ~]$ ssh-keygen -t rsa
```

(**passphrase** 입력하라는 문구가 뜨면 그냥 계속 엔터)

```
[bigdata@server01 ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/bigdata/.ssh/id_rsa):
Created directory '/home/bigdata/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/bigdata/.ssh/id_rsa.
Your public key has been saved in /home/bigdata/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:/aeEaEb78qzXDFuqRu7Ao1TbCkSDAA92SMmH0nC/Is0 bigdata@server01
The key's randomart image is:
+---[RSA 2048]-----+
|0==.
|oXo+
|. + +
| 0 . 0 .
|. E o . S .
|. 0 0 +.0.0.
| 0 =0* .Bo .
|. 0 *+0+.00
|. .0+*+ .
+---[SHA256]-----+
```

19) 생성된 SSH 공개키를 다른 서버에 복사

```
[bigdata@server01 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server02
```

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

bigdata@server02's password: 에 비밀번호 **1111** 입력

```
[bigdata@server01 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server02
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/bigdata/.ssh/id_rsa.pub"
The authenticity of host 'server02 (192.168.56.102)' can't be established.
ECDSA key fingerprint is SHA256:657fD+iUU4guvjfvxQ/QAUEJ9tV8wa5fTTiaRglyxvw.
ECDSA key fingerprint is MD5:f3:ff:b8:02:1b:98:a2:e0:36:98:f9:6d:86:52:2b:a1.
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 the new keys
bigdata@server02's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'bigdata@server02'"
and check to make sure that only the key(s) you wanted were added.
```

## 20) 네임노드 서버에서 데이터 노드로 SSH 접속 시도

```
[bigdata@server01 ~]$ ssh server02
```

```
[bigdata@server01 ~]$ ssh server02
Last login: Wed May 23 15:20:50 2018 from gateway
[bigdata@server02 ~]$
```

### ○ 다른 데이터 노드들에게도 모두 실행

```
[bigdata@server01 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server01
[bigdata@server01 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server03
[bigdata@server01 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server04
```

### ○ server02에 bigdata 계정으로 접속해 18~20번 똑같이 실행

```
[bigdata@server02 ~]$ ssh-keygen -t rsa
```

```
[bigdata@server02 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server01
[bigdata@server02 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server02
[bigdata@server02 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server03
[bigdata@server02 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server04
```



- **server03**에 **bigdata** 계정으로 접속해 18~20번 똑같이 실행

```
[bigdata@server03 ~]$ ssh-keygen -t rsa
```

```
[bigdata@server03 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server01  
[bigdata@server03 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server02  
[bigdata@server03 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server03  
[bigdata@server03 ~]$ ssh-copy-id -i /home/bigdata/.ssh/id_rsa.pub bigdata@server04
```

- 프로토콜 버퍼를 다운 받아 파일질라를 통해 **/usr/local**에 업로드
  - 프로토콜 버퍼는 구글에서 공개한 오픈소스 직렬화 라이브러리
  - 하둡2는 내부 데몬간의 데이터 통신을 위해 프로토콜 버퍼 적용

21) root 계정으로 Server01에 접속해 /usr/local로 이동

```
[root@server01 ~]# cd /usr/local
```

22) protobuf-2.5.0.tar.gz 압축 풀기

```
[root@server01 local]# tar xvfz protobuf-2.5.0.tar.gz
```

23) 프로토콜 버퍼는 c,c++ 기반이기 때문에 c와 c++의 컴파일러 설치

```
[root@server01 local]# yum -y install gcc gcc-c++
```

#### 24) 다음과 같이 설치를 진행

```
[root@server01 local]# cd protobuf-2.5.0  
[root@server01 protobuf-2.5.0]# ./configure  
[root@server01 protobuf-2.5.0]# make  
[root@server01 protobuf-2.5.0]# make install
```

#### 25) 설치가 완료되면 protoc 명령어를 실행해 버전 확인

```
[root@server01 protobuf-2.5.0]# protoc --version
```

```
[root@server01 protobuf-2.5.0]# protoc --version  
libprotoc 2.5.0
```

#### 26) 다른 서버에 모두 프로토콜 버퍼 설치(x3)

#### 27) 앞으로 wget으로 프로그램을 다운받기 위해 yum으로 wget 다운로드

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

## ■ 하둡2 - 네임노드 HA 설치

호스트 명	하둡2 설치 내용
server01	주키퍼, 액티브 네임노드, 저널노드, 데이터노드, 리소스매니저
server02	주키퍼, 스탠바이 네임노드, 저널노드, 데이터노드, 노드매니저
server03	주키퍼, 저널노드, 데이터노드, 노드매니저
server04	데이터노드, 노드매니저

1) **server01**에 **bigdata**계정으로 로그인 후 주키퍼 다운로드

- 혹은 filezilla로 업로드

```
$ wget http://mirror.navercorp.com/apache/zookeeper/stable/zookeeper-3.4.12.tar.gz
```

2) 주키퍼 파일 압축 풀기

```
$ tar xvfz zookeeper-3.4.12.tar.gz
```

3) 주키퍼 설정파일이 있는 “/home/bigdata/zookeeper-3.4.12/conf” 로 이동

```
$ cd zookeeper-3.4.12/conf
```

4) 주키퍼 설정을 하기 위해 샘플 설정 파일인 “zoo\_sample.cfg”를 “zoo.cfg” 이란 이름으로 복사

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

5) zoo.cfg를 vi 에디터로 열어 아래와 같이 입력후 저장(:wq)

```
$ vi zoo.cfg
```

#주키퍼의 스냅샷 경로 설정(var/zookeeper폴더는 아래에서 생성할 예정)

```
dataDir=/home/bigdata/var/zookeeper
```

#주키퍼의 최대 클라이언트 커넥션 개수이며 0은 무제한을 의미함

```
maxClientCnxns=0
```

#세션별 타임아웃 시간이며 ms단위로 설정

```
maxSessionTimeout=180000
```

#멀티서버로 구성할 서버 등록으로 2888은 주키퍼의 리더에 접속하는 포트이고 3888은 리더를 결정하는 데 사용 server.# 뒤의 값은 해당 서버의 아이디

```
server.1=server01:2888:3888
```

```
server.2=server02:2888:3888
```

```
server.3=server03:2888:3888
```

```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
syncLimit=5
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
dataDir=/home/bigdata/var/zookeeper
# 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=server01:2888:3888
server.2=server02:2888:3888
server.3=server03:2888:3888
```

6) /home/bigdata/var/zookeeper 디렉토리 생성

```
[bigdata@server01 ~]$ cd /home/bigdata  
[bigdata@server01 ~]$ mkdir var  
[bigdata@server01 ~]$ mkdir var/zookeeper
```

7) "sever.#"에서 서버별로 부여한 아이디를 주키퍼 데이터 디렉터리에도 저장

```
[bigdata@server01 ~]$ cd var/zookeeper  
[bigdata@server01 ~]$ vi myid
```

8) myid에 아이디 입력 후 저장(:wq)

```
[bigdata@server01 zookeeper]$ cat myid  
1
```

9) scp와 ssh명령어를 이용해서 다른 서버(server02, server03)에도 주키퍼 설치  
- SSH에서 다른 서버로 파일을 전송 하고자 할때 SCP를 이용하여 전송

```
[bigdata@server01 ~]$ cd ~  
[bigdata@server01 ~]$ tar cvfz zookeeper.tar.gz zookeeper-3.4.12  
[bigdata@server01 ~]$ scp zookeeper.tar.gz bigdata@server02:/home/bigdata  
[bigdata@server01 ~]$ scp zookeeper.tar.gz bigdata@server03:/home/bigdata
```

10) server02와 server03 각각의 서버에 bigdata 계정으로 로그인해서  
- 압축파일 풀기

```
[bigdata@server02 ~]$ tar xvfz zookeeper.tar.gz
```

- /home/bigdata/var/zookeeper 디렉토리 생성

```
[bigdata@server02 ~]$ cd ~  
[bigdata@server02 ~]$ mkdir var  
[bigdata@server02 ~]$ mkdir var/zookeeper
```

```
[bigdata@server02 ~]$ cd var/zookeeper
```

```
[bigdata@server02 ~]$ vi myid
```

```
[bigdata@server02 zookeeper]$ cat myid  
2
```

```
[bigdata@server03 zookeeper]$ cat myid  
3
```

11) 주키퍼를 설치한 서버마다(server01, server02, server03) bigdata 계정으로 접속 후 주키퍼 파일 디렉토리로 이동 후 주키퍼 실행

```
$ cd zookeeper-3.4.12/
```

```
$ ./bin/zkServer.sh start
```

12) 주키퍼 상태 확인

```
$ cd zookeeper-3.4.12/
```

```
$ ./bin/zkServer.sh status
```

```
[bigdata@server01 zookeeper-3.4.12]$ ./bin/zkServer.sh status  
ZooKeeper JMX enabled by default  
Using config: /home/bigdata/zookeeper-3.4.12/bin/./conf/zoo.cfg  
Mode: follower
```

```
[bigdata@server02 zookeeper-3.4.12]$ ./bin/zkServer.sh status  
ZooKeeper JMX enabled by default  
Using config: /home/bigdata/zookeeper-3.4.12/bin/./conf/zoo.cfg  
Mode: leader
```

```
[bigdata@server03 zookeeper-3.4.12]$ ./bin/zkServer.sh status  
ZooKeeper JMX enabled by default  
Using config: /home/bigdata/zookeeper-3.4.12/bin/./conf/zoo.cfg  
Mode: follower
```