

하둡 클러스터 구축 과정 및 매뉴얼

작성자: 송원문 (moonie@onycom.com)

작성일: 2014.08.20. (최종 업데이트)

진행 순서

1. 서버 OS 설치
2. 하둡 클러스터 운영을 위한 기본 환경 설정
3. 하둡 클러스터 설치 및 설정 변경
4. 하둡 클러스터 운영
5. 하둡 에코시스템 설치
6. 하둡 퍼포먼스 튜닝

※ 주의:

본 설치 매뉴얼은, 다수의 서버로 구성된 멀티 클러스터 환경에서 하둡 시스템을 구축하기 위한 방법을 기술한 것으로, 단일 노드에서의 하둡 환경 구축 과정은 본 매뉴얼의 내용과 다르게 진행되어야 함.

1. 서버 OS 설치

■ DVD를 이용한 OS 설치 과정 (각 서버에 아래 모든 과정을 동일하게 반복 수행)

- 1) CentOS 6.5버전 ISO 다운로드 및 DVD 준비
- 2) DVD를 이용한 서버 설치
- 3) 미디어 테스트 과정 Skip (DVD 내용 검증 과정)
- 4) 언어는 한국어로 선택
- 5) 저장장치 대상은 "기본저장장치"로 진행
- 6) 호스트 명 및 네트워크 정보 설정(호스트 설정 화면 좌측 하단)
 - 호스트 명은, 실제 네트워크에서 사용할 호스트 명으로 설정
: 이후의 하둡 설치 과정에서 그대로 사용되어야 함
 - 네트워크 정보 설정시, "자동으로 연결" 체크 선택 하여 진행
- 7) root 계정 암호 설정
- 8) "사용자 레이아웃 만들기" 선택하여 파티션 직접 설정
(본 설치에서는 SSD 250GB와 HDD 1TB가 설치되어 있다고 가정)
 - SSD 중, 100GB를 swap로 마운트
: 메모리 영역으로, 속도가 빠른 SSD에 할당
 - SSD의 남은 가용 전체 용량을 '/'로 마운트 (OS용도 목적)
 - HDD의 전체 용량을 '/home'으로 마운트
: HDD는 모든 사용자 데이터 및 HDFS로 활용 목적
: 단 향후, HDD의 확장이 가능해야 하는 경우라면 LVM을 이용하여 설정)
- 9) 부트로더는 기본 옵션으로 진행
- 10) 설치 형태는 Desktop 기본 모드로 설치 진행 (단일 서버로 OS 설치 목적)

■ 기초 설치 후, 설정 과정 (각 서버에 아래 모든 과정을 동일하게 반복 수행)

- 1) 사용자 추가시, 하둡 운영을 위한 사용자 계정 추가
 - 본 설치에서는 hadoop-user의 계정을 추가
 - 각 서버에서의 사용자별 패스워드는 동일하게 설정
- 2) 시간 설정시, 네트워크로 동기화 선택
 - 여러대의 하둡 클러스터의 시스템 시간 동기화를 위한 목적
 - 네트워크 서버 선택은 가장 첫 번째로 제공되는 서버 선택
- 3) 사용자 자동 로그인 설정
 - 원격 리부팅시, 로그인 이후 구동되는 서비스의 자동 시작을 위한 목적
 - 본 설치에서는 root로 자동 로그인하도록 설정
 - /etc/gdm/custom.conf 파일에 다음의 내용 추가

[daemon]	<-- 이 부분에 아래 내용 추가
AutomaticLoginEnable=true	<-- 이 부분 추가
AutomaticLogin=root	<-- 이 부분 추가

- 4) 터미널을 이용한 ssh 접속 테스트

- 기존 키 정보 때문에 접속이 되지 않는 경우 다음 방법으로 기존 키 정보 삭제

ssh-keygen -R ip-address (또는 hostname)
--

2. 서버 기본 환경 설정

- 호스트 및 자바 운영 환경 설정 (각 서버에 아래 모든 과정을 동일하게 반복 수행)

1) 서버 운영을 위한 호스트 파일 정보 추가 (root로 로그인 하여 설정)

- /etc/hosts 파일의 내용에 다음과 같은 내용 추가

: 네트워크 주소 및 호스트명은, 각 서버의 OS 설치시 적용했던 내용으로 입력 반드시, 실제 호스트명과 일치해야 하며 그렇지 않은 경우, 하둡 미작동

```
# hadoop cluster
192.168.x.xxx    onymaster
192.168.x.xxx    onydata1
192.168.x.xxx    onydata2
192.168.x.xxx    onydata3
192.168.x.xxx    onydata4
192.168.x.xxx    onydata5
```

2) 자바 설치 (root로 로그인 하여 설정)

- 시스템 운용시 Java 1.6 버전 이상의 정식 JDK를 설치하여 사용

: CentOS 6.5 버전인 경우, OpenJDK가 기본으로 설치됨

: 알려진 바는 없으나, OpenJDK와 관련된 버그 발생을 사전 차단하기 위함

: Hadoop 1.x 이하 버전은 Java 1.6버전 이상에서 정상 구동된다고 알려져 있음

- 시스템에 설치된 Java 버전 확인

```
# rpm -qa | grep java
java-1.7.0-openjdk-1.7.0.45-2.4.3.3.el6.x86_64      // openjdk
tzdata-java-2013g-1.el6.noarch                     // openjdk 의존 추정
java-1.6.0-openjdk-1.6.0.0-1.66.1.13.0.el6.x86_64  // openjdk
```

: 확인 결과, OpenJDK 또는 별도의 Java가 설치되어 있지 않은 경우

> 다음 페이지 ①과정 없이, 바로 ②과정으로 진행

: 확인 결과, Java 1.6 버전 미만이 설치되어 있는 경우

> 다음 페이지 ①과정을 참고하여 기존 Java를 삭제하고, ②과정 진행

[자바 설치 과정 계속]

① 기존 JDK 삭제

- : 본 설치에서는 OpenJDK를 삭제 하지 않고, 사용하지 않는 것으로 설정 변경
- : CentOS 6.5버전 설치시에는 '/usr/bin/java'에 OpenJDK 기본 설치되어 있음

```
# mv /usr/bin/java /usr/bin/java-old
```

② 최신 버전의 JDK 설치

- : 본 설치에서는 oracle java 7버전을 다운로드하여 이용함
- : 다음의 웹 주소에서, tar.gz 형태로 제공되는 java 7 최신 버전을 다운로드

```
http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html
```

- : 다운로드 받은 'jdk-7u67-linux-x64.gz'파일을 서버에 업로드하여 설치
- 본 설치에서는, root 홈 디렉토리에 먼저 파일을 복사하고
- > 압축을 해제 한 후 -> 시스템 폴더로 이동하고자 함

```
# tar -zxvf jdk-7u67-linux-x64.gz          // 압축 해제
-- 출력 생략 (종료 후, 'jdk1.7.0_67' 폴더 생성) --
# mv jdk1.7.0_67 /usr/local                // 폴더 이동
```

- > 압축 해제 후, 시스템 폴더로 이동하지 않아도 되나
- 본 설치에서는, 관리 편의상, 시스템 폴더로 이동함
- : 환경 변수 등록하고, 반영하기
- 다음을 참고하여 '/etc/profile' 파일의 제일 뒤에, 내용을 추가 하고 반영

```
# vi /etc/profile          // 환경 변수 등록
-- 내용 생략, 파일의 맨 뒤에 아래 내용 추가 --
export JAVA_HOME=/usr/local/jdk1.7.0_67
export CLASSPATH=.:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin
-- 저장 후, vi 종료
# source /etc/profile      // 환경 변수 반영
# java -version            // 설치된 java 버전 확인
```

- 하둡 클러스터 운영을 위한 ssh 공개키 교환 설정 (hadoop-user로 로그인 하여 설정)
 - : 하둡은, ssh 프로토콜을 통하여 클러스터 내 서버들 간에 통신을 수행
 - : ssh 공개키를 설정하지 않으면, hadoop내 ssh 접속마다 패스워드를 요구함
 - : 따라서, ssh 공개키를 설정하여, hadoop내 통신을 원활하게 해야 함
 - 공개키는 dsa와 rsa가 있으며, 선택적 사용 가능 (rsa가 보안이 높다고 알려짐)
 - : ssh 통신을 편하게 사용하기 위하여 초기 설치시 모든 서버에 동일 계정 생성
 - 본 설치에서는 'hadoop-user'계정을 생성하였음

1) ssh 공개키 교환 방법

- ① 각 서버에서 공개 키 생성
- ② 생성된 공개키를 하둡 클러스터 노드들간에 서로 공유(복사)
 - 단, 본 설치에서는 편의를 위해 다음과 같은 단계로 진행
- ① 각 서버에서 공개 키 생성 (rsa 방식)
- ② master node에서 각 서버들에서의 공개키를 복사해서 통합 공개키 생성
- ③ master node의 통합 공개키를 각 서버에 복사

2) 하둡 master node로 사용할 서버에 하둡 사용 계정으로 로그인

- 본 설치에서는 'hadoop-user'계정을 사용

3) 다음 명령을 이용하여 서버별 공개 키 생성 (반드시, 계정 홈디렉토리에서 진행)

- 모든 서버에서 동일하게 작업을 완료 한 후, 다음 단계로 진행 필요
 - : 반드시, 하둡 사용 계정으로 로그인 하여 작업 수행
- 생성 단계에서, 생성 위치와 패스워드는 모두 입력 없이 <Enter>
 - : 기본 위치에 생성하도록, 확인 절차 없이 접속하도록 하기 위함

```
# ssh-keygen -t rsa           // rsa 타입 공개키 생성
-- 이전 생략 --
Enter file in which to save the key (/home/hadoop-user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
-- 이후 생략 --
# ls .ssh                     // 생성 확인
id_rsa id_rsa.pub
```

4) master node에서 공개키 통합 파일 생성 (본 작업은 master node에서만 수행)

- 생성된 공개키를 이용하여 서버별 접속의 인증에 사용할 파일
- 자신의 공개키를 포함한 통합 파일 생성

```
# cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

- : 생성된 '~/.ssh/authorized_keys'파일은 소유자 외 쓰기 권한이 없어야 함
 - (모든 서버에 생성된 파일이 권한이 동일해야 함)
 - 그렇지 않은 경우, ssh 접속을 하더라도 암호를 물어보게 됨

- 타 서버의 공개키를 복사해서 상기 생성한 통합 파일에 내용을 추가
: 모든 data node 서버의 공개키를 복사하여 통합 파일 내용 업데이트

```
# ssh hadoop-user@onydata1 cat ~/.ssh/id_rsa.pub >> .ssh/authorized_keys
The authenticity of host 'onydata1 (192.168.1.126)' can't be established.
RSA key fingerprint is 75:b9:02:ce:4a:8d:0b:0a:e3:ce:c5:a2:09:b4:94:90.
Are you sure you want to continue connecting (yes/no)? => yes 입력
Warning: Permanently added 'onydata1,192.168.1.126' (RSA) to the list of
known hosts.
hadoop-user@onydata1's password: => 해당 서버의 접속 패스워드 입력
# ssh hadoop-user@onydata2 cat ~/.ssh/id_rsa.pub >> .ssh/authorized_keys
==> 상기 과정과 동일하게 진행
# ssh hadoop-user@onydata3 cat ~/.ssh/id_rsa.pub >> .ssh/authorized_keys
==> 상기 과정과 동일하게 진행
# ssh hadoop-user@onydata4 cat ~/.ssh/id_rsa.pub >> .ssh/authorized_keys
==> 상기 과정과 동일하게 진행
# ssh hadoop-user@onydata5 cat ~/.ssh/id_rsa.pub >> .ssh/authorized_keys
==> 상기 과정과 동일하게 진행
```

- 5) 통합된 공개키 인증 파일을 각 data node 서버로 복사
: scp명령을 이용하여 모든 data node 서버에 파일을 복사

```
# scp -rp .ssh/authorized_keys hadoop-user@onydata1:~/.ssh/
hadoop-user@onydata1's password: => 해당 서버의 접속 패스워드 입력
authorized_keys                  100% 2413    2.4KB/s   00:00
# scp -rp .ssh/authorized_keys hadoop-user@onydata2:~/.ssh/
==> 상기 과정과 동일하게 진행
# scp -rp .ssh/authorized_keys hadoop-user@onydata3:~/.ssh/
==> 상기 과정과 동일하게 진행
# scp -rp .ssh/authorized_keys hadoop-user@onydata4:~/.ssh/
==> 상기 과정과 동일하게 진행
# scp -rp .ssh/authorized_keys hadoop-user@onydata5:~/.ssh/
==> 상기 과정과 동일하게 진행
```

- 6) 공개키를 이용한 ssh 자동 접속 확인
- 각 서버간에 ssh 접속이 비밀번호 없이 연결되면 성공

```
# ssh hadoop-user@onydata1
```

3. 하둡 클러스터 설치 및 설정 변경

■ 하둡 다운로드 및 설치 및 운영 준비 (hadoop-user로 로그인 하여 설정)

- : 각 서버에 하둡 파일을 다운로드 받아 압축을 해제하고 경로 설정
- : 하둡 설치 및 운영을 위한 각종 디렉토리 경로 생성
- : 반드시 하둡을 운영할 계정으로 각 서버에 접속하여 본 작업을 수행
- : 본 작업은, 모든 서버에서 동일한 작업을 수행

1) 다음의 링크에서 하둡 1.0.3 버전을 다운로드

- 'hadoop-1.0.3.tar.gz' 파일을 다운로드

- : 하둡은 버전이 여러 개가 존재하나, 본 설치에서는 하둡 1.0.3버전을 설치
- : hadoop 설치 이후, ankus를 설치/사용하기 위한 목적
- : ankus는 hadoop 1.0.3버전에서만, 모든 개발 기능에 대한 테스트가 진행되었음

<http://archive.apache.org/dist/hadoop/core/hadoop-1.0.3/>

2) 다운로드 받은 하둡 파일을 서버에 업로드 하고 압축 해제

- 하둡 사용자 계정의 홈 디렉토리에 업로드 하고, 압축 해제
- 편리한 사용을 위한 심볼릭 링크 생성 (향후 버전 변경 및 패스 관리 등)

```
# tar -zxvf hadoop-1.0.3.tar.gz          // 압축 해제
-- 압축 해제 --
# ln -s hadoop-1.0.3 hadoop             // 심볼릭 링크 생성
# ls -l                                 // 최종 확인
lrwxrwxrwx. 1 hadoop-user users      12 2014-08-19 14:45 hadoop ->
hadoop-1.0.3
drwxr-xr-x. 14 hadoop-user users    4096 2012-05-09 05:35 hadoop-1.0.3
-rw-r--r--. 1 hadoop-user users 62428860 2014-08-19 14:33
hadoop-1.0.3.tar.gz
```

3) 하둡 수행을 위한 HADOOP_HOME을 생성하고 경로를 PATH에 추가

- 전체 적용을 위하여, root로 변경 접속하여 수행

```
# su root
passwd: (root 패스워드 입력)
# vi /etc/profile                      // 환경 변수 등록
-- 내용 생략, 앞서 java 설치시 추가 내용을 다음과 같이 추가 변경 --
export JAVA_HOME=/usr/local/jdk1.7.0_67
export CLASSPATH=.:$JAVA_HOME/lib/tools.jar
export HADOOP_HOME=/home/hadoop-user/hadoop
export PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin
-- 상기 2줄 참고하여 추가/변경 --
-- 저장 후, vi 종료 --
# exit                                // su 명령 종료
# source /etc/profile                  // 'hadoop-user'계정에 환경 변수 반영
```

- 4) 하둡 운영을 위한 기본 저장소 경로 생성 (모든 서버에서 동일한 작업 수행)
- 하둡을 운영할 계정(hadoop-user)의 홈 디렉토리에 hdfs 라는 폴더를 생성
: 관리 및 설정 편의 목적상, 모든 서버에 동일한 위치, 이름으로 경로 생성
: 하둡 운영 저장소 경로는, 목적에 따라, 별도의 디렉토리로 해도 무관
: 설치 폴더는 'rwxr-xr-x'권한이 설정되어야 함
 - 다음과 같이 경로를 이동/확인하고, 기본 디렉토리를 생성

```
# cd ~                // 현재 위치를 hadoop-user의 홈디렉토리로 이동
# pwd                 // 현재 위치 확인
/home/hadoop-user
# mkdir ./hdfs        // 현재 위치에 hdfs라는 디렉토리를 생성
```

- 5) 상세 항목별 저장소 경로 생성 (모든 서버에서 동일한 작업 수행)
- 하둡의 임시 파일, 데이터 저장 등 각종 파일들의 저장 위치 설정
: 설치 폴더는 'rwxr-xr-x'권한이 설정되어야 함
 - 설정하지 않는 경우, 하둡 소스내에서 지정한 기본 값으로 지정
 - 이 설정이 필요 없는 경우, 상기 4) 과정도 필요 없음
 - 단, 본 설치에서는 별도 관리를 위해 파티션도 별도 설정 하고, 본 단계를 진행

```
# mkdir hdfs/tmp       // 하둡 임시 파일 저장 위치
# mkdir hdfs/namenode  // 하둡 네임 노드 정보 저장 위치
# mkdir hdfs/datanode  // 하둡 데이터 노드 정보 저장 위치
```

- 6) 하둡 프로세스 정보 저장 경로 생성 (모든 서버에서 동일한 작업 수행)
- 하둡은 서버 시작시, 프로세스 정보를 특정 경로에 저장
 - 하둡 서버 종료시 저장된 프로세스 정보를 이용하여 프로세스 종료 수행
 - 기본적으로는 리눅스의 /tmp 디렉토리를 사용하나 다음과 같은 문제 발생 가능
: 이 폴더는 오래된 파일을 자동 삭제하도록 OS에서 관리하는 폴더
: 하둡 프로세스 정보 파일이 삭제되는 경우, 하둡 종료가 되지 않는 경우 발생
 - 이러한 현상을 방지하기 위하여 별도의 하둡 프로세스 정보 저장 경로를 생성
: 본 설치에서는 다음과 같은 경로를 생성하여 활용

```
# mkdir hadoop/pids    // 하둡 임시 파일 저장 위치
```


■ 하둡 설정 파일 내용 업데이트 (네임노드 서버에서 hadoop-user로 로그인 하여 설정)

- 하둡 환경 설정 파일의 정보 업데이트
- 상기 생성한 디렉토리 정보를 이용하여 하둡 환경 설정 파일 정보를 업데이트
- 설정한 환경 정보를 데이터 노드 서버에 복사

1) 'core-site.xml' 파일 정보 업데이트

```
# vi hadoop/conf/core-site.xml
<property>
  <name>fs.default.name</name>           // HDFS 기본 위치/포트
  <value>hdfs://onymaster:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>           // 하둡 작업 임시 디렉토리
  <value>/home/hadoop-user/hdfs/tmp</value>
</property>
```

2) 'hdfs-site.xml' 파일 정보 업데이트

```
# vi hadoop/conf/hdfs-site.xml
<property>
  <name>dfs.name.dir</name>           // 하둡 네임노드 정보 위치
  <value>/home/hadoop-user/hdfs/namenode</value>
</property>
<property>
  <name>dfs.data.dir</name>           // 하둡 데이터노드 정보 위치
  <value>/home/hadoop-user/hdfs/datanode</value>
</property>
<property>
  <name>dfs.replication</name>       // 데이터 복제 개수
  <value>3</value>
</property>
```

3) 'mapred-site.xml' 파일 정보 업데이트

```
# vi hadoop/conf/mapred-site.xml
<property>
  <name>mapred.job.tracker</name>      // 잡트래커 위치/포트
  <value>onymaster:9001</value>
</property>
<property> // 태스크트래커의 맵 동시실행 개수 (cpu 개수 고려 설정)
  <name>mapred.tasktracker.map.tasks.maximum</name>
  <value>4</value> // 기본 값은 2
</property>
<property> // 태스크트래커의 리듀스 동시실행 개수 (cpu 개수 고려 설정)
  <name>mapred.tasktracker.reduce.tasks.maximum</name>
  <value>3</value> // 기본 값은 2
</property>
```

4) 'masters' 파일 정보 수정

- SecondaryNameNode로 사용할 서버의 호스트 정보 기입
- 본 설치에서는 기본 네임노드에서 같이 운영하므로, 네임 노드 정보 기입

```
# vi hadoop/conf/masters
onymaster
```

- SecondaryNameNode가 네임노드와 다른 경우, 본 4) 과정 이전의 1)~3)에서 SecondaryNameNode를 위한 추가 설정이 필요함
(본 매뉴얼에서는 포함하지 않음)

5) 'slaves' 파일 정보 수정

- 하둡 클러스터의 데이터 노드로 사용할 서버들의 호스트 정보 기입

```
# vi hadoop/conf/slaves
onydata1
onydata2
onydata3
onydata4
onydata5
```

6) 'hadoop-env.sh' 파일 정보 수정

- 하둡 운영을 위한 java 경로 정보 수정
- 'hadoop-env.sh' 파일에서 '# export JAVA_HOME=/usr~~~'부분을 수정
: 주석을 해제 하고, java가 설치된 디렉토리 정보로 수정

```
# vi hadoop/conf/hadoop-env.sh
-- 생략 --
export JAVA_HOME=/usr/local/jdk1.7.0_67
-- 생략 --
```

- 하둡 프로세스 정보 저장 경로 설정
- 'hadoop-env.sh' 파일에서 '# export HADOOP_PID_DIR=/var~~~' 부분을 수정
: 주석을 해제 하고, 전 단계에서 생성한 디렉토리 정보로 수정

```
# vi ~/hadoop/conf/hadoop-env.sh
-- 중략 --
export HADOOP_PID_DIR=/home/hadoop-user/hadoop/pids
-- 중략 --
```

7) 데이터 노드 서버에 환경 설정 파일 복사

- 모든 하둡 클러스터는 동일한 환경으로 설정하여 운영하고자 함

```
# scp -rp ./hadoop/conf/* hadoop-user@onydata1:~/hadoop/conf/  
# scp -rp ./hadoop/conf/* hadoop-user@onydata2:~/hadoop/conf/  
# scp -rp ./hadoop/conf/* hadoop-user@onydata3:~/hadoop/conf/  
# scp -rp ./hadoop/conf/* hadoop-user@onydata4:~/hadoop/conf/  
# scp -rp ./hadoop/conf/* hadoop-user@onydata5:~/hadoop/conf/
```

8) 네임노드 서버 및 분산 파일 시스템 초기화

- 하둡 클러스터 운영을 위해 HDFS 파일 시스템을 포맷하고 준비

```
# hadoop namenode -format  
-- 생략 --  
Re-format filesystem in /home/hadoop-user/hdfs/namenode ? (Y or N) => Y  
-- 생략 --
```

4. 하둡 클러스터 운영

■ 하둡 수행 및 확인

- 모든 설치가 종료되면, 하둡 클러스터를 구동

1) 하둡 클러스터 구동 (네임노드 서버에서 hadoop-user로 로그인 하여 설정)

- 'hadoop/bin/start-all.sh'파일을 이용하여 하둡 클러스터를 구동

```
# hadoop/bin/start-all.sh
starting namenode, logging to
/home/hadoop-user/hadoop-1.0.3/libexec/../logs/hadoop-hadoop-user-nameno
de-onymaster.out
onydata5: starting datanode, logging to
/home/hadoop-user/hadoop-1.0.3/libexec/../logs/hadoop-hadoop-user-datanod
e-onydata5.out
-- 중략 --
onymaster: starting secondarynamenode, logging to
/home/hadoop-user/hadoop-1.0.3/libexec/../logs/hadoop-hadoop-user-secondar
ynamenode-onymaster.out
starting jobtracker, logging to
/home/hadoop-user/hadoop-1.0.3/libexec/../logs/hadoop-hadoop-user-jobtrack
er-onymaster.out
onydata5: starting tasktracker, logging to
/home/hadoop-user/hadoop-1.0.3/libexec/../logs/hadoop-hadoop-user-tasktrack
er-onydata5.out
```

2) 하둡 클러스터 정상 구동 확인

- 네임 노드 서버에서 'jps'명령을 이용하여 다음과 같은 프로세스를 확인
: 단, 프로세스 번호는 다를 수 있으며, 최소 다음과 같은 프로세스 확인

```
# jps
14660 SecondaryNameNode
14761 JobTracker
14899 Jps
14464 NameNode
```

- 각각의 데이터노드 서버에서 'jps'명령을 이용하여 다음과 같은 프로세스를 확인
: 단, 프로세스 번호는 다를 수 있으며, 최소 다음과 같은 프로세스 확인

```
# jps
6176 Jps
6088 TaskTracker
5969 DataNode
```

3) 하둡 구동 확인

- 하둡 클러스터가 정상적으로 구동되면, 다음과 같은 두 개의 URL 접속 확인
: 네임노드 서버와 포트를 지정하여 확인
- 하둡 잡트래커 메인 페이지 > <http://onymaster:50070>

onymaster Hadoop Map/Reduce Administration

State: RUNNING
 Started: Tue Aug 19 17:26:29 KST 2014
 Version: 1.0.3, r1335192
 Compiled: Tue May 8 20:31:25 UTC 2012 by hortonfo
 Identifier: 201408191726

Cluster Summary (Heap Size is 480 MB/889 MB)

Running Map Tasks	Running Reduce Tasks	Total Submissions	Nodes	Occupied Map Slots	Occupied Reduce Slots	Reserved Map Slots	Reserved Reduce Slots	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node	Blacklisted Nodes	Graylisted Nodes
0	0	0	5	0	0	0	0	10	10	4.00	0	0

Scheduling Information

Queue Name	State	Scheduling Information
default	running	N/A

Filter (Jobid, Priority, User, Name)
 Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

Running Jobs

Retired Jobs

- 하둡 DFS 메인 페이지 > <http://onymaster:50030>

NameNode 'onymaster:9000'

Started: Tue Aug 19 17:26:29 KST 2014
 Version: 1.0.3, r1335192
 Compiled: Tue May 8 20:31:25 UTC 2012 by hortonfo
 Upgrades: There are no upgrades in progress.

[Browse the filesystem](#)
[Namenode Logs](#)

Cluster Summary

8 files and directories, 1 blocks = 9 total. Heap Size is 480 MB / 889 MB (53%)

Configured Capacity	: 4.01 TB
DFS Used	: 176 KB
Non DFS Used	: 210.66 GB
DFS Remaining	: 3.8 TB
DFS Used%	: 0 %
DFS Remaining%	: 94.87 %
Live Nodes	: 5
Dead Nodes	: 0
Decommissioning Nodes	: 0
Number of Under-Replicated Blocks	: 0

NameNode Storage:

Storage Directory	Type	State
/home/hadoop-user/hdfs/namenode	IMAGE_AND_EDITS	Active

This is [Apache Hadoop](#) release 1.0.3

4) 방화벽 설정 확인

- 하둡 클러스터가 제대로 수행되지 않는 경우, 방화벽 문제인 경우가 있음
- CentOS 6.5의 경우, 기본 방화벽 수행이 설정되어 있음
 - : 일부 지정된 포트만을 허용하도록 방화벽 수행이 기본 설정되어 있음
- 하둡은 여러개의 별도의 포트를 사용하므로, 방화벽을 해제하여, 포트를 열어줌
- 단, 내부 클러스터인 경우, 외부 접속이 불가능 하므로 모든 방화벽을 해제
 - : 외부 클러스터인 경우, 사용 포트만 열어 주는 것이 타당
- 모든 서버에서 다음을 수행하여 방화벽을 해제하고, 본 단계의 1)~3)을 재 수행

```
# su root
passwd: (root 패스워드 입력)
# service iptables stop          // 방화벽 적용 중지
# chkconfig iptables off        // 리부팅시 방화벽 자동 시작 금지
```

5) 경고 메시지 해결

- 설치 후, 'hadoop fs -ls /'등의 하둡 명령 실행시 다음과 같은 경고 발생 가능
 - : Warning: \$HADOOP_HOME is deprecated
 - : 하둡 1.x 버전에서 나타나는 bug, 하둡 운영에는 오류가 없음
- 이 경우, 하둡 설정 파일 중 'hadoop-env.sh' 파일을 수정하여 경고 회피 가능
 - : 모든 서버에 있는 'hadoop-env.sh' 파일을 동일하게 수정해야 함
 - : 하둡 구동중인 경우, 하둡 구동을 중지하고, 수정 후 재 구동 필요
 - (하둡 구동 중지는 네임노드에서 '~/.hadoop/bin/stop-all.sh'을 실행하여 수행)

```
# vi ~/.hadoop/conf/hadoop-env.sh    // 하둡 설정 파일 수정
export HADOOP_HOME_WARN_SUPPRESS="TRUE"
-- 파일의 맨 뒤에, 상기 내용을 추가하여 저장 --
```

■ MapReduce 작업 수행 확인

- 설치된 하둡 클러스터에서 MapReduce 작업이 정상적으로 수행되는지 확인
- 하둡 설치 파일에서 제공되는 예제 중, WordCount 예제를 수행하여 테스트
 - : 파일내에 나타난 단어별 횟수를 계산하는 MapReduce 알고리즘
- 아래의 작업은, 네임노드 서버에서 진행

1) 테스트 파일 생성 및 서버 업로드

- 다음과 같은 텍스트 파일을 생성하여 서버에 업로드 ('test.txt')

```
hadoop is so good
hello hadoop
this is test file for hadoop
```

2) 테스트 파일을 HDFS에 업로드

```
# hadoop fs -mkdir /data // hdfs에 데이터 업로드 폴더 생성
# hadoop fs -ls / // 폴더 생성 확인
drwxr-xr-x - hadoop-user supergroup 0 2014-08-19 18:00 /data
drwxr-xr-x - hadoop-user supergroup 0 2014-08-19 16:56 /home
# hadoop fs -copyFromLocal ./test.txt /data/ // 테스트 파일 업로드
# hadoop fs -ls /data // 파일 업로드 확인
-rw-r--r-- 3 hadoop-user supergroup 61 2014-08-19 18:02 /data/test.txt
```

3) WordCount 예제 수행

- HDFS상의 입력 데이터 파일과 출력 파일 생성 경로 지정하여 예제 수행
- 예제를 위한 파일은 hadoop 디렉토리에 위치하고 있음

```
# cd hadoop
# hadoop jar hadoop-examples-1.0.3.jar wordcount /data/test.txt /wc
-- 결과 생략 --
```

4) WordCount 예제 수행 결과 확인

```
# hadoop fs -ls / // 지정한 출력 경로 생성 확인
Found 3 items
drwxr-xr-x - hadoop-user supergroup 0 2014-08-19 18:02 /data
drwxr-xr-x - hadoop-user supergroup 0 2014-08-19 16:56 /home
drwxr-xr-x - hadoop-user supergroup 0 2014-08-19 19:00 /wc
# hadoop fs -ls /wc // 출력 경로내 결과 파일 생성 확인
Found 3 items
-rw-r--r-- 3 hadoop-user supergroup 0 2014-08-19 19:00 /wc/_SUCCESS
drwxr-xr-x - hadoop-user supergroup 0 2014-08-19 19:00 /wc/_logs
-rw-r--r-- 3 hadoop-user supergroup 61 2014-08-19 19:00 /wc/part-r-00000
# hadoop fs -cat /wc/part-r-00000 // 출력 경로내 결과 파일 내용 확인
file 1
for 1
good 1
hadoop 3
hello 1
is 2
so 1
test 1
this 1
```


5. 하둡 에코시스템 설치

- 스크립트 기반 데이터 분석 환경인 Pig 설치
- SQL 기반 데이터 분석 환경인 Hive 설치

6. 하둡 퍼포먼스 튜닝

- 임시 파일 및 입출력 파일에 대한 압축 허용 설정, 압축 라이브러리 등