

# 하둡 클러스터 개요 및 설치

순천향대학교 컴퓨터공학과  
이 상 정

하둡 클러스터 개요 및 설치

## 학습 내용

1. 하둡 클러스터 개요
2. 하둡 설치

---

## 1. 하둡 클러스터 개요

하둡 클러스터 개요 및 설치

### 클러스터와 분산 시스템

---

#### □ 클러스터 (Cluster)

- 독립된 컴퓨터들의 네트워크
- 각 컴퓨터는 자신의 메모리와 운영체제를 가짐
- 입출력 연결망(interconnection)으로 연결
  - E.g., Ethernet/switch, Internet
- 컴퓨터 구조 관점

#### □ 분산 시스템 (Distributed System)

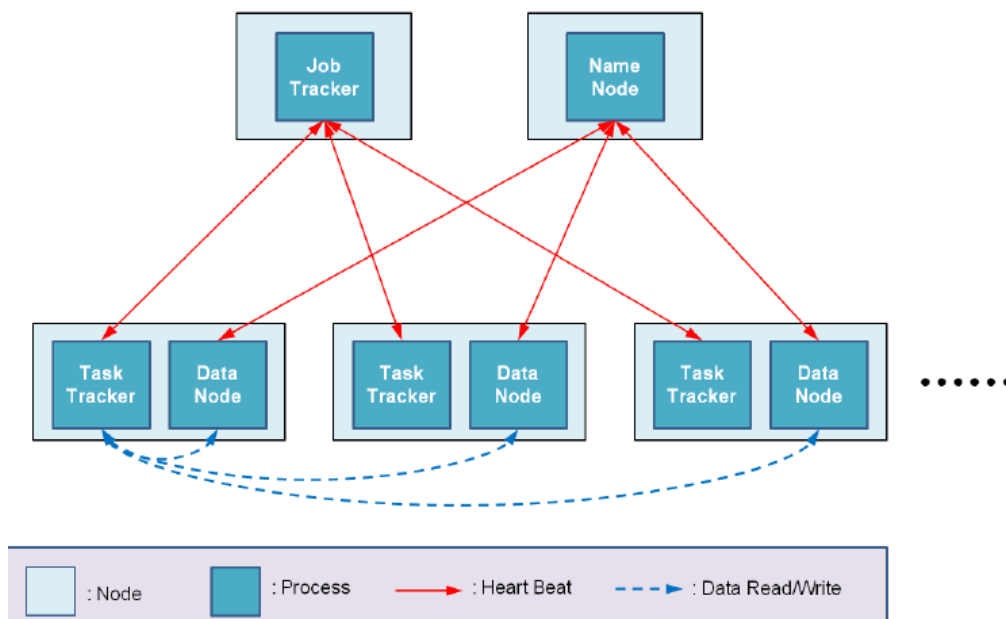
- 대용량 데이터를 처리/관리하거나 혹은 시간이 매우 오래 걸리는 복잡한 계산 작업을 수행하기 위해 **여러 대의 컴퓨터를 네트워크로 연결**해서 **하나의 시스템**처럼 사용할 수 있도록 구성한 시스템
- 운영체제 관점

## 하둡 주요 구성 요소

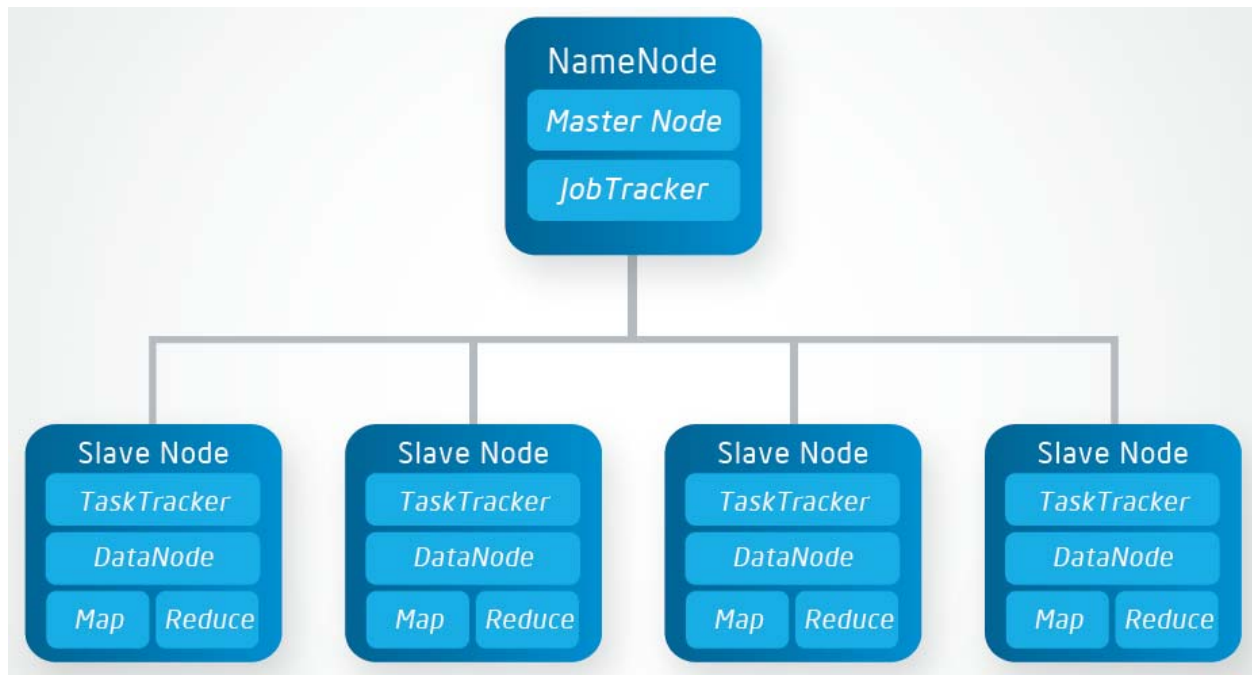
- 분산 파일 시스템인 HDFS(Hadoop Distributed File System)와 분산 처리 시스템인 맵리듀스로 구성
  - 마스터/슬레이브(Master/Slave) 구조로 구성
  - HDFS에서 마스터는 네임 노드(name node), 슬레이브는 데이터 노드(data node)
  - 맵리듀스에서 마스터는 JobTracker, 슬레이브는 TaskTracker
- HDFS의 마스터/슬레이브 동작
  - 마스터인 네임 노드가 파일의 메타(meta) 정보를 관리하고, 실제 데이터는 여러 대의 데이터 노드에 분산해서 저장
  - 데이터는 일정 크기(디폴트 64MB)의 블록단위로 나뉘어 관리
  - 블록들을 여러 대의 데이터 노드에 분산 및 복제해서 저장

## 하둡 시스템 아키텍처

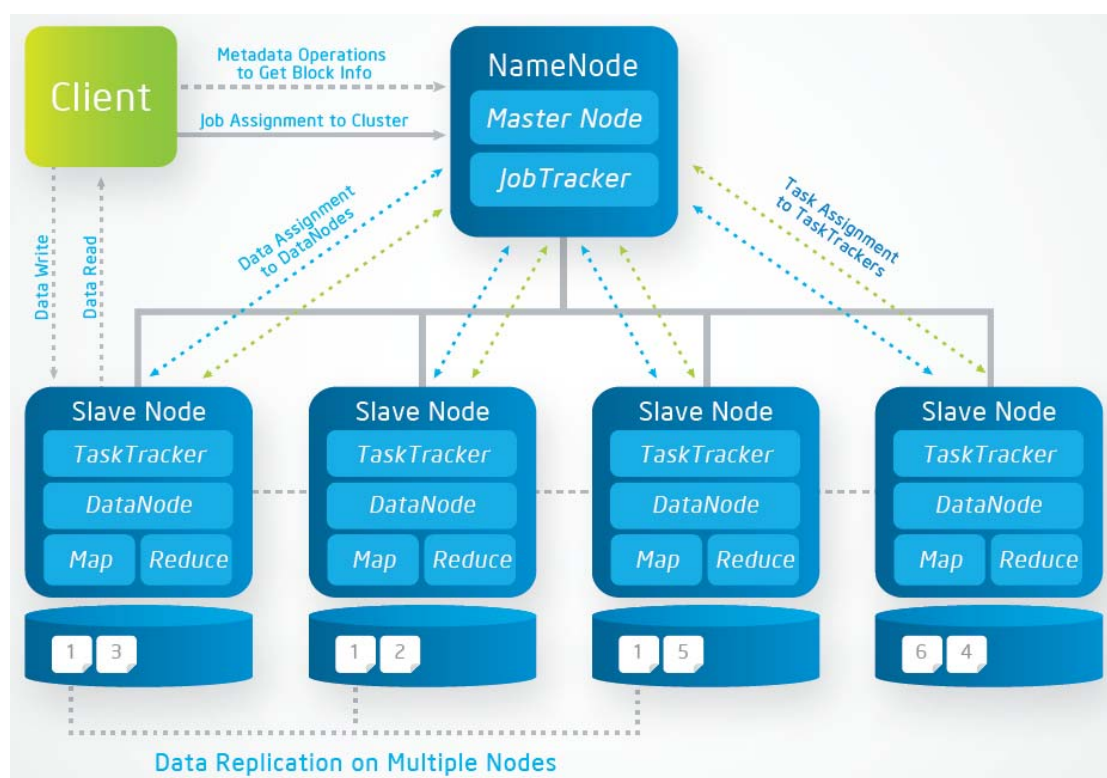
Hadoop System Architecture



## 하둡 클러스터 구성 예



## 하둡 클러스터 동작 예



## HDFS 소개

- HDFS(Hadoop Distributed File System)는 하나의 저장 장치에 저장이 어려운 대규모 데이터를 높은 가용성을 유지하면서 관리하기 위한 목적으로 설계
  - HDFS(Hadoop Distributed File System) 는 파일을 트리 형식의 디렉토리 구조로 관리
    - 파일 혹은 디렉토리의 복사, 이동, 삭제 및 권한 설정 등의 기능을 제공
    - 기존의 일반 파일 시스템과 유사 기능

## HDFS 특징 (1)

- 일반 파일 시스템과는 구분되는 다음과 같은 특징을 가짐
  - 데이터 블록 (Data Block)
    - HDFS는 파일을 블록단위로 분할하고 여러 데이터 노드에 분산 저장
    - 각 파일의 기본 정보(이름, 디렉토리, 권한, 기타 등등) 및 각 블록들의 위치 정보를 네임 노드에서 관리
    - 파일을 블록 단위로 분할하여 관리하기 때문에 하나의 파일이 수 테라 바이트 정도로 크더라도 관리가 가능
    - 보통 하나의 블록은 수십 ~ 수백 메가 바이트 정도로 설정
  - 복제 (Replication)
    - 일부 데이터 노드에 장애 발생 시 데이터 손실을 막기 위해 각 데이터 블록에 대해서 여러 개의 복제본(Replica)을 가짐
    - 복제본의 개수는 파일별로 직접 사용자가 지정해 줄 수 있으며 직접 지정하지 않으면 보통 3개의 복제본을 유지
    - 네임 노드는 데이터 노드들과 주기적으로 통신하면서 데이터 노드의 상태를 감시. 이렇게 상태 확인을 위해 주고 받는 패킷을 Heart beat라고 함

## HDFS 특징 (2)

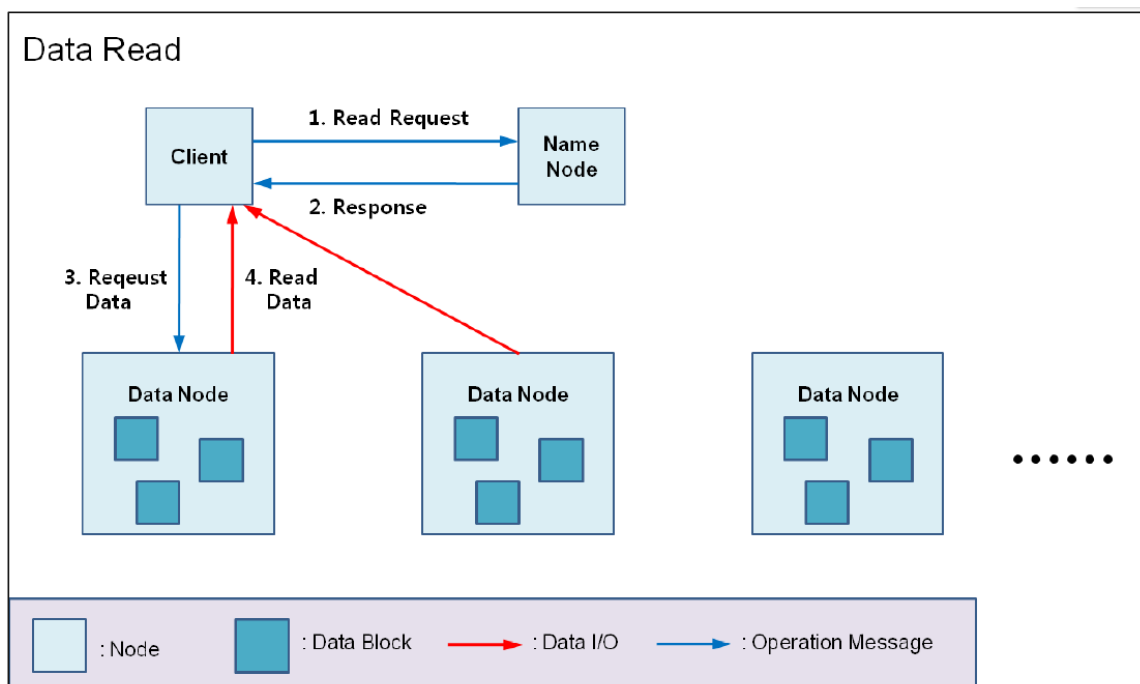
- 랙 인지(Rack Awareness)

- 전기 문제 혹은 통신 두절 등의 문제로 인한 **랙(Rack: 서버들을 여러 개 묶어서 관리하는 단위)** 혹은 센터 단위로 문제 발생 시에도 가용성을 높이기 위해 데이터 블록의 복제본을 관리
- 전체 클러스터의 **위상 구조(Topology)**를 토대로 복제본이 한 군데에 몰려 있지 않도록 관리
- 예를 들어 복제본 개수가 3인 경우 두 개는 같은 랙의 다른 노드에 저장하고 나머지 하나는 다른 랙에 있는 노드에 저장

- 데이터 읽기: 지역성 (Locality)

- 사용자가 파일을 읽을 때는 먼저 **네임 노드에 해당 파일 위치 정보를 요청**하고 그 정보를 토대로 데이터 노드와 통신
- 네임 노드는 읽기 요청을 한 노드와 가장 가까운 데이터 노드 순으로 정렬해서 알려주기 때문에 사용자는 **가장 가까운 데이터 노드에서 블록 정보**를 가져오게 됨

## 데이터 읽기

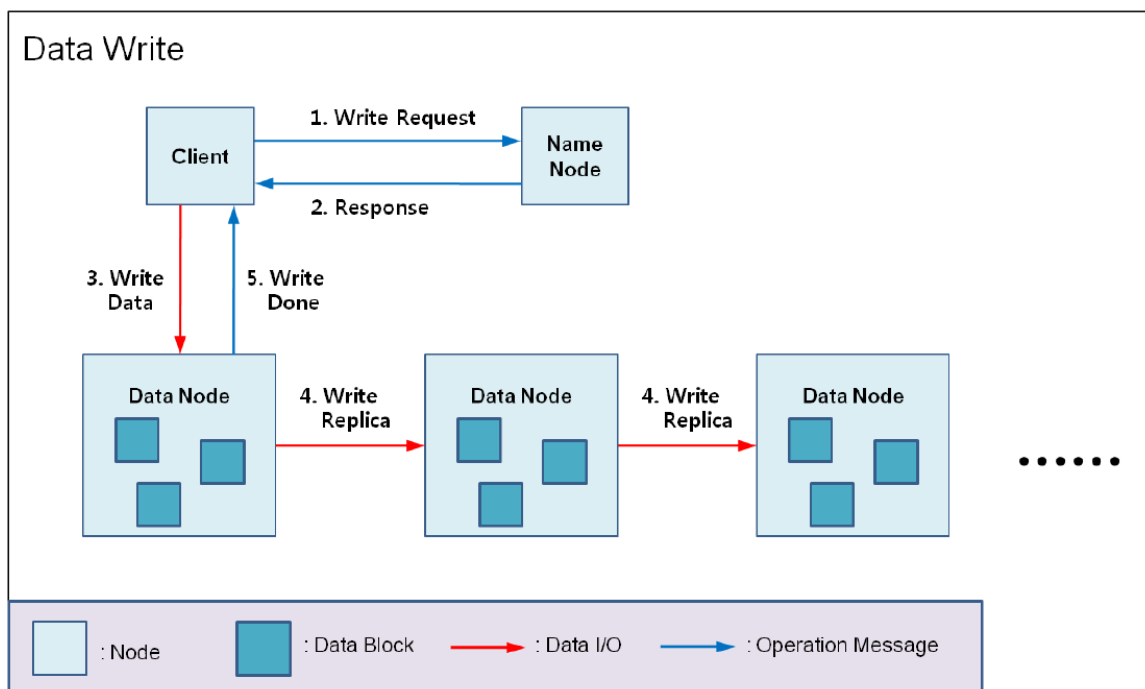


## HDFS 특징 (3)

- 데이터 쓰기: 일관성(Consistency)

- 사용자가 파일을 쓸 때는 먼저 네임 노드에 해당 파일 쓰기 요청을 함
- 네임 노드는 해당 파일이 이미 있는지, 사용자가 쓰기 권한을 갖고 있는지 등의 기본적인 유효성 검사를 수행
- 이 검사가 완료되면 네임 노드는 데이터를 저장할 데이터 노드 리스트를 사용자에게 전달
- 사용자는 이 중 첫 번째 데이터 노드에 데이터를 쓰기 시작하고, 이 데이터 노드는 다른 데이터 노드들과 파이프 라인처럼 연결돼 **순차적으로 복제 데이터의 쓰기 수행**
- 사용자는 첫 번째 데이터 노드에 쓰기 작업이 완료되더라도 모든 복제본 생성이 완료될 때까지 대기

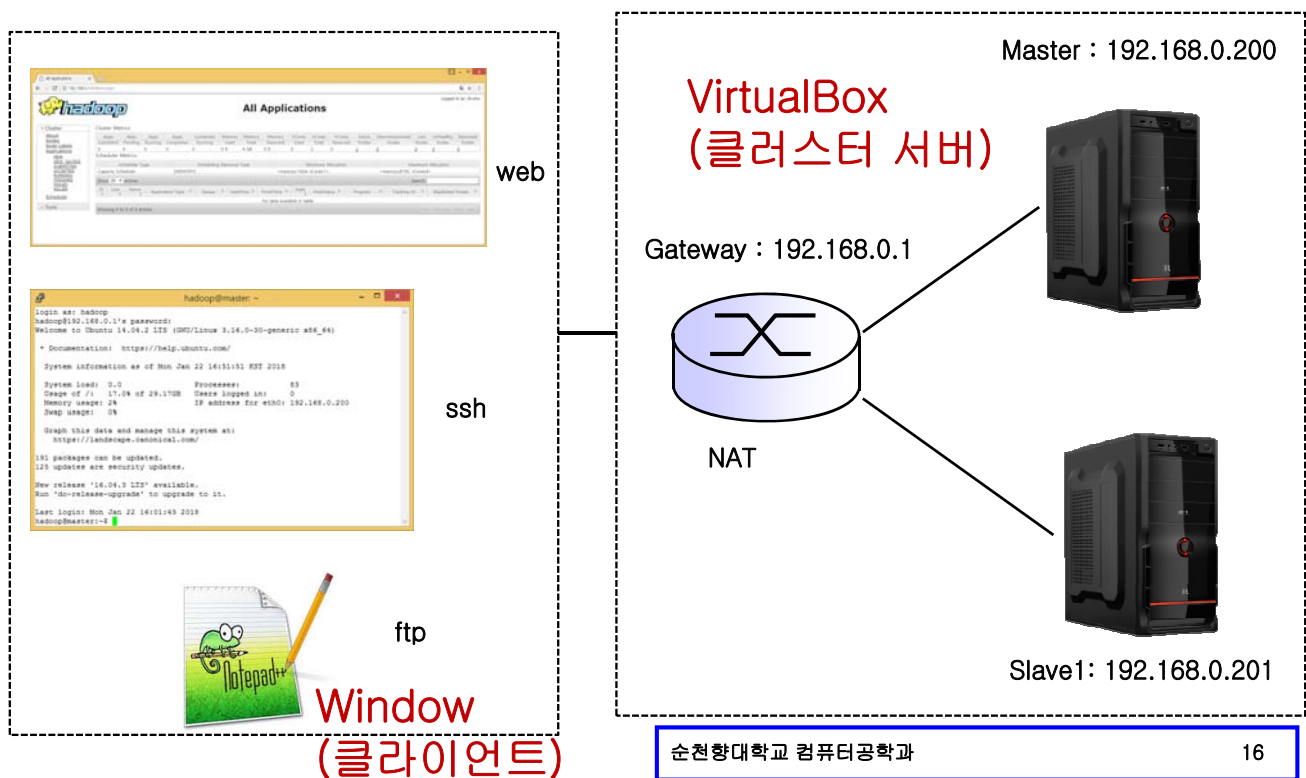
## 데이터 쓰기



## 2. 하둡 설치

### 하둡 클러스터 개요 및 설치

## 하둡 클러스터 네트워크 환경





## 하둡 클러스터 구성 요소

### 가상머신 서버 IP 및 하둡 노드 기능

| 번호 | Host Name | IP            | Hadoop                       |
|----|-----------|---------------|------------------------------|
| 1  | master    | 192.168.0.200 | Namenode, Datanode           |
| 2  | slave1    | 192.168.0.201 | Secondary Namenode, Datanode |

### 가상머신 하드웨어 사양

| 번호 | Host Name | OS                 | RAM | HDD  |
|----|-----------|--------------------|-----|------|
| 1  | master    | Ubuntu 18.04.2 LTS | 4GB | 40GB |
| 2  | slave1    | Ubuntu 18.04.2 LTS | 4GB | 40GB |

## 오라클 자바 8 설치 (1)

### 오라클의 자바 라이선스 정책 변경으로 인해 ppa 폐기

- Oracle 홈페이지의 tar.gz 형태 JDK 파일 수동 다운로드
- <https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

**Java SE Development Kit 8u211**

You must accept the [Oracle Technology Network License Agreement](#) for Oracle Java SE to download this software.

☐ Accept License Agreement
 ☒ Decline License Agreement

| Product / File Description  | File Size | Download  |
|-----------------------------|-----------|---|
| Linux ARM 32 Hard Float ABI | 72.86 MB  | <a href="#">jdk-8u211-linux-arm32-vfp-hflt.tar.gz</a> |
| Linux ARM 64 Hard Float ABI | 69.76 MB  | <a href="#">jdk-8u211-linux-arm64-vfp-hflt.tar.gz</a> |
| Linux x86                   | 174.11 MB | <a href="#">jdk-8u211-linux-i586.rpm</a>              |
| Linux x86                   | 188.92 MB | <a href="#">jdk-8u211-linux-i586.tar.gz</a>           |
| Linux x64                   | 171.13 MB | <a href="#">jdk-8u211-linux-x64.rpm</a>               |
| Linux x64                   | 185.96 MB | <a href="#">jdk-8u211-linux-x64.tar.gz</a>            |

## Oracle Java 8 설치 (2)

### □ 모든 노드에 자바 8 설치

- 압축 해제 후 이동

```
bigdata@master: ~
bigdata@master:~$ ls
hadoop-2.7.7          hbase-2.2.0-bin.tar.gz  spark-2.4.3-bin-hadoop2.7
hadoop-2.7.7.tar.gz  jdk-8u201-linux-x64.tar.gz  spark-2.4.3-bin-hadoop2.7.tgz
bigdata@master:~$
```

```
$ tar -xvzf jdk-8u201-linux-x64.tar.gz
$ sudo mkdir /usr/lib/jvm
$ sudo mv jdk1.8.0_201 /usr/lib/jvm
```

- Java 환경변수 설정
  - ~/.bashrc 에 추가

```
export JAVA_HOME="/usr/lib/jvm/jdk1.8.0_201"
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=$JAVA_HOME/lib:$CLASSPATH
```

## Oracle Java 8 설치 (3)

- 추가 후 적용
  - \$ source ~/.bashrc
- 자바 심볼릭 링크 생성

```
$sudo update-alternatives --install "/usr/bin/java" "java" "/usr/lib/jvm/jdk1.8.0_201/bin/java" 1
$sudo update-alternatives --install "/usr/bin/javac" "javac" "/usr/lib/jvm/jdk1.8.0_201/bin/javac" 1
$sudo update-alternatives --install "/usr/bin/javaws" "javaws" "/usr/lib/jvm/jdk1.8.0_201/bin/javaws" 1
```

- 재 부팅
  - \$ sudo reboot

- 자바 버전 확인
  - \$ java -version

```
bigdata@master:~$ java -version
java version "1.8.0_201"
Java(TM) SE Runtime Environment (build 1.8.0_201-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
```

## 노드의 호스트 이름 설정

### 모든 노드에 호스트 이름 설정

- 하둡의 마스터 노드와 슬레이브 노드의 호스트 이름과 IP 지정
- 파일명 : **/etc/hosts**

```

.....

ff02::2 ip6-allrouters
ff02::3 ip6-allhosts

192.168.0.200 master
192.168.0.201 slave1
.....

```

## 가상머신 자바 및 네트워크 테스트

```

bigdata@master: ~
Last login: Wed Jul 10 07:51:48 2019 from 192.168.0.1
bigdata@master:~$
bigdata@master:~$ java -version
java version "1.8.0_201"
Java(TM) SE Runtime Environment (build 1.8.0_201-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
bigdata@master:~$
bigdata@master:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:83:0e:85 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.200/24 brd 192.168.0.255 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe83:e85/64 scope link
        valid_lft forever preferred_lft forever
bigdata@master:~$

bigdata@slave1: ~
Last login: Wed Jul 10 08:04:38 2019
bigdata@slave1:~$
bigdata@slave1:~$ java -version
java version "1.8.0_201"
Java(TM) SE Runtime Environment (build 1.8.0_201-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
bigdata@slave1:~$
bigdata@slave1:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:a1:e3:e2 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.201/24 brd 192.168.0.255 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe81:e3e2/64 scope link
        valid_lft forever preferred_lft forever
bigdata@slave1:~$

```

## 하둡 설치 - 패키지 다운로드

### □ 마스터 노드에서 설치한 후에 슬레이브 노드에 설치 및 설정 복사

- 지금부터 마스터 노드에 설치 과정 소개
  - 마스터 설치 후에 슬레이브에 설치한 디렉토리 전체 복사

### □ 패키지 다운로드 후 압축 해제

- 하둡 2.7.7 다운로드
  - 2019년 7월 기준 스파크 2.4버전 사용 시 안정된 버전인 2.7.7 사용
  - <http://hadoop.apache.org/releases.html>

```
$ wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.7/hadoop-2.7.7.tar.gz
```

- 압축 해제

```
$ tar -xvzf hadoop-2.7.7.tar.gz
```

## 하둡 설정 - 환경 변수

### □ 하둡 환경 변수 설정

- ~/.bashrc 에 추가

```
export HADOOP_PREFIX=$HOME/hadoop-2.7.7
export HADOOP_HOME=$HOME/hadoop-2.7.7
export PATH=$PATH:$HADOOP_PREFIX/bin
export PATH=$PATH:$HADOOP_PREFIX/sbin
export HADOOP_MAPRED_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_HOME=${HADOOP_PREFIX}
export HADOOP_HDFS_HOME=${HADOOP_PREFIX}
export YARN_HOME=${HADOOP_PREFIX}
export HADOOP_YARN_HOME=${HADOOP_PREFIX}
export HADOOP_CONF_DIR=${HADOOP_PREFIX}/etc/hadoop
export YARN_CONF_DIR=${HADOOP_PREFIX}/etc/hadoop
```

```
#Native Path
export HADOOP_COMMON_LIB_NATIVE_DIR=${YARN_HOME}/lib/native
export HADOOP_OPTS="-Djava.library.path=${YARN_HOME}/lib"
```

### □ 추가 후 적용

```
$ source ~/.bashrc
```

## 하둡 설정 - 데이터 노드 설정

### □ 데이터 노드 설정

- 2개의 노드를 모두를 데이터 노드로 지정
- 파일명: `~/hadoop-2.7.7/etc/hadoop/slaves`

```
master
slave1
```

## 하둡 설정 - SSH 인증 설정

### □ 마스터 노드에서 슬레이브 노드로 암호 없는 SSH 접속을 위해 모든 노드의 홈 디렉토리 아래 `.ssh` 디렉토리에 SSH 키 생성

- 마스터 노드에서 각 노드의 공용 키(public key) 생성

```
$ ssh-keygen -t rsa
$ ssh slave1 'ssh-keygen -t rsa'
```

- 각 노드에 생성된 공용 키를 `~/ssh/authorized_keys`에 수집

```
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
$ ssh slave1 'cat ~/.ssh/id_rsa.pub' >> ~/.ssh/authorized_keys
```

- 마스터에서 수집된 공용 키를 슬레이브 노드로 배포

```
$ scp ~/.ssh/authorized_keys slave1:~/.ssh/authorized_keys
```

- 테스트: 암호 입력 요구 없이 접속 해야 성공

```
$ ssh slave1
```

## 하둡 설정 - core-site.xml

### □ 네임 노드의 URI 설정

- HDFS와 맵리듀스에 공통으로 사용되는 하둡 코어를 위한 환경 설정
- fs.default.name 항목이 네임노드를 가리키면 하둡의 맵리듀스 작업은 자동으로 HDFS를 이용하여 입력파일 전달
- ~/hadoop-2.7.7/etc/hadoop/core-site.xml

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://master:9000</value>
  </property>
</configuration>
```

## 하둡 설정 - hdfs-site.xml

### □ 네임노드, 보조 네임 노드, 데이터 노드 등과 같은 HDFS 데몬을 위한 환경 설정

- ~/hadoop-2.7.7/etc/hadoop/hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>3</value>
  </property>
  <property>
    <name>dfs.name.dir</name>
    <value>/home/bigdata/hadoop-2.7.7/hdfs/name</value>
  </property>
  <property>
    <name>dfs.data.dir</name>
    <value>/home/bigdata/hadoop-2.7.7/hdfs/data</value>
  </property>
  <property>
    <name>dfs.http.address</name>
    <value>master:50070</value>
  </property>
  <property>
    <name>dfs.permissions</name>
    <value>>false</value>
  </property>
  <property>
    <name>dfs.namenode.secondary.http-address</name>
    <value>slave1:50090</value>
  </property>
</configuration>
```

## 하둡 설정 - yarn-site.xml (1)

- YARN의 자원 관리자, 노드 관리자를 위한 환경 설정
  - `~/hadoop-2.7.7/etc/hadoop/yarn-site.xml`

```
<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.resourcemanager.resource-tracker.address</name>
    <value>master:8025</value>
  </property>
  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>master:8030</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>master:8050</value>
  </property>
</configuration>
```

29

## 하둡 설정 - yarn-site.xml (2)

```
<property>
  <name>yarn.resourcemanager.webapp.address</name>
  <value>master:8088</value>
</property>
<property>
  <name>yarn.nodemanager.resource.memory-mb</name>
  <value>2048</value>
</property>
<property>
  <name>yarn.nodemanager.resource.cpu-vcores</name>
  <value>1</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>
```

## 하둡 설정 - mapred-site.xml

### □ 맵리듀스의 JobTracker, TaskTracker 데몬을 위한 환경 설정

- ~/hadoop-2.7.7/etc/hadoop/mapred-site.xml

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

## 하둡 설정 - hadoop-env.sh

### □ hadoop-env.sh

- 하둡을 구동하는 스크립트에서 사용되는 환경 변수 설정
  - ~/hadoop-2.7.7/etc/hadoop/hadoop-env.sh

```
export JAVA_HOME="/usr/lib/jvm/jdk1.8.0_201"
export HADOOP_HOME=PREFIX=${HADOOP_HOME}
.....
```



## 슬레이브 노드에 하둡 배포

### □ 마스터 노드의 설치 및 설정을 슬레이브 노드에 배포

- .bashrc 배포

```
$ scp /home/bigdata/.bashrc slave1:~
```

- 하둡 설치 디렉토리 배포

```
$ scp -r /home/bigdata/hadoop-2.7.7 slave1:~
```

## 하둡 실행 – 초기 실행 (1)

### □ 마스터 노드에서 하둡 초기 실행

- 모든 하둡 클러스터의 HDFS 시작, YARN 데몬 실행
  - `$ start-all.sh`
    - 종료 시에는 `stop-all.sh`
  - !! 실습 종료 시에는 반드시 마스터에서 하둡 종료  
`$ stop-all.sh`
- 단, 처음 실행 시에 **한번 만** 마스터에서 네임 노드를 포맷
  - `$ hadoop namenode -format`

```
$ start-all.sh  
$ hadoop namenode -format
```

- 참고: 하둡 HDFS 시작, YARN 데몬 실행
  - `$ start-dfs.sh`
  - `$ start-yarn.sh`
  - 종료 시에는 `stop-dfs.sh, stop-yarn.sh`

## 하둡 실행 - 초기 실행 (2)

```
bigdata@master:~$ start-all.sh
```

```
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [master]
master: starting namenode, logging to /home/bigdata/hadoop-2.7.7/logs/hadoop-big
data-namenode-master.out
slave1: starting datanode, logging to /home/bigdata/hadoop-2.7.7/logs/hadoop-big
data-datanode-slave1.out
master: starting datanode, logging to /home/bigdata/hadoop-2.7.7/logs/hadoop-big
data-datanode-master.out
Starting secondary namenodes [slave1]
```

```
bigdata@master:~$ hadoop namenode -format
```

```
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

19/07/12 01:02:52 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = master/192.168.0.200
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 2.7.7
STARTUP_MSG: classpath = /home/bigdata/hadoop-2.7.7/etc/hadoop:/home/bigdata/h
adoop-2.7.7/share/hadoop/common/lib/junit-4.11.jar:/home/bigdata/hadoop-2.7.7/sh
```

## 하둡 클러스터 동작 확인

### □ 마스터와 슬레이브 노드 동작 확인

- 하둡 버전 확인, **\$ hdfs version**
- 프로세스 확인, **\$ jps**

```
bigdata@master:~$ hdfs version
```

```
Hadoop 2.7.7
Subversion Unknown -r c1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac
Compiled by stevel on 2018-07-18T22:47Z
Compiled with protoc 2.5.0
From source with checksum 792e15d20b12c74bd6f19a1fb886490
This command was run using /home/bigdata/hadoop-2.7.7/share/hadoop/common/hadoop-common-2.7.7.jar
```

```
bigdata@master:~$ jps
```

```
21184 ResourceManager
21377 NodeManager
21576 Jps
20937 DataNode
20732 NameNode
bigdata@master:~$
```

```
bigdata@slave1:~$ jps
```

```
13875 Jps
13415 DataNode
13593 SecondaryNameNode
13743 NodeManager
bigdata@slave1:~$
```

## 클러스터 동작 확인 - 네임노드 웹 (1)

### 네임노드 웹 접속하여 확인

- <http://192.168.0.1:50070>

Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

### Overview 'master:9000' (active)

|                |  |
|----------------|--|
| Started:       | Thu Jul 11 03:02:06 UTC 2019                     |
| Version:       | 2.7.7, rc1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac |
| Compiled:      | 2018-07-18T22:47Z by stevel from branch-2.7.7    |
| Cluster ID:    | CID-ba53327c-f0bc-47d1-a14b-0ad7874daf29         |
| Block Pool ID: | BP-951907935-192.168.0.100-1562813557274         |

### Summary

Security is off.  
Safemode is off.  
1 files and directories, 0 blocks = 1 total filesystem object(s).

순천향대학교 컴퓨터공학과
37

## 클러스터 동작 확인 - 네임노드 웹 (2)

Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

### Datanode Information

#### Datanode usage histogram

#### In operation

| Node                                  | Last contact | Admin State | Capacity | Used  | Non DFS Used | Remaining | Blocks | Block pool used | Failed Volumes | Version |
|---------------------------------------|--------------|-------------|----------|-------|--------------|-----------|--------|-----------------|----------------|---------|
| master:50010<br>(192.168.0.200:50010) | 1            | In Service  | 18.57 GB | 32 KB | 9.96 GB      | 7.65 GB   | 0      | 32 KB (0%)      | 0              | 2.7.7   |
| slave1:50010<br>(192.168.0.201:50010) | 2            | In Service  | 18.57 GB | 32 KB | 7.26 GB      | 10.35 GB  | 0      | 32 KB (0%)      | 0              | 2.7.7   |

38

## 하둡 실행 테스트 – wordcount (1)

### □ 하둡 명령 형식

- ```
$ hadoop fs -command [arg ...]
```
- ```
$ hdfs dfs -command [arg ...]
```
- command: ls, cat, cp, mv, put, get, ....

### □ 테스트 예

- HDFS 루트에 input 디렉토리 생성
- 로컬 파일 README.txt를 HDFS /input 에 복사
- wordcount 예 실행
- 실행 결과 출력

```
$ hadoop fs -ls /
$ hadoop fs -mkdir /input
$ hadoop fs -put /home/bigdata/hadoop-2.7.7/README.txt /input
$ hadoop fs -ls /input
$ hadoop jar /home/bigdata/hadoop-2.7.7/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.7.jar wordcount /input/README.txt /output
$ hadoop fs -cat /output/part-r-00000
```

순천향대학교 컴퓨터공학과

39

## 하둡 실행 테스트 – wordcount (2)

```
bigdata@slave1:~$ hadoop fs -ls /
bigdata@slave1:~$ hadoop fs -mkdir /input
bigdata@slave1:~$ hadoop fs -ls /
Found 1 items
drwxr-xr-x - bigdata supergroup          0 2019-07-12 01:16 /input
bigdata@slave1:~$
```

```
bigdata@slave1:~$ hadoop fs -put /home/bigdata/hadoop-2.7.7/README.txt /input
bigdata@slave1:~$ hadoop fs -ls /input
Found 1 items
-rw-r--r--  3 bigdata supergroup      1366 2019-07-12 01:17 /input/README.txt
bigdata@slave1:~$
```

## 하둡 실행 테스트 - wordcount (3)

```
bigdata@slave1:~$ hadoop jar /home/bigdata/hadoop-2.7.7/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.7.jar wordcount /input/README.txt /output
19/07/12 01:18:32 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
19/07/12 01:18:32 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
19/07/12 01:18:33 INFO input.FileInputFormat: Total input paths to process : 1
19/07/12 01:18:33 INFO mapreduce.JobSubmitter: number of splits:1
19/07/12 01:18:33 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1633203680_0001
19/07/12 01:18:33 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
19/07/12 01:18:33 INFO mapreduce.Job: Running job: job_local1633203680_0001
19/07/12 01:18:33 INFO mapred.LocalJobRunner: OutputCommitter set in config null

      HDFS: Number of write operations=1
Map-Reduce Framework
  Map input records=31
  Map output records=179

      total committed heap usage (bytes)=553003304
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=1366
File Output Format Counters
  Bytes Written=1306
bigdata@slave1:~$
```

## 하둡 실행 테스트 - wordcount (3)

```
bigdata@slave1:~$ hadoop fs -ls /output
Found 2 items
-rw-r--r-- 3 bigdata supergroup 0 2019-07-12 01:18 /output/_SUCCESS
-rw-r--r-- 3 bigdata supergroup 1306 2019-07-12 01:18 /output/part-r-00000
bigdata@slave1:~$
bigdata@slave1:~$ hadoop fs -cat /output/part-r-00000
(BIS), 1
(ECCN) 1
(TSU) 1
(see 1
5D002.C.1, 1
740.13) 1
<http://www.wassenaar.org/> 1
Administration 1
Apache 1
BEFORE 1
BIS 1
Bureau 1
Commerce, 1
Commodity 1
Control 1
Core 1
Department 1
ENC 1
Exception 1
Export 2
```

## YARN 자원관리자 확인

## □ YARN 자원 관리자(resource manager) 확인

- <http://192.168.0.1:8088>



**hadoop All Applications**

Cluster Metrics

| 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 |
|----------------|--------------|--------------|----------------|--------------------|-------------|--------------|-----------------|-------------|--------------|-----------------|--------------|----------------------|
| 1              | 0            | 0            | 1              | 0                  | 0 B         | 8 GB         | 0 B             | 0           | 8            | 0               | 1            | 0                    |

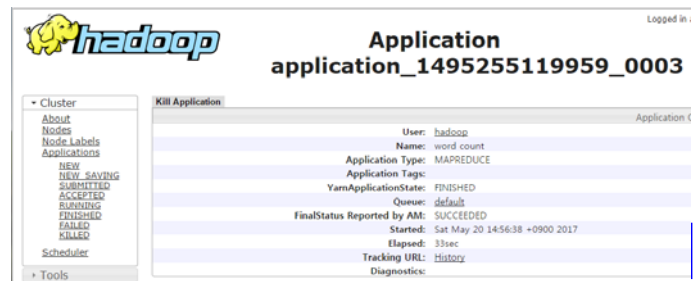
Scheduler Metrics

| Scheduler Type     | Scheduling Resource Type | Minimum Allocation      |
|--------------------|--------------------------|-------------------------|
| Capacity Scheduler | [MEMORY]                 | <memory:1024, vCores:1> |

Show 20 entries

| ID                             | User   | Name       | Application Type | Queue   | StartTime                      | FinishTime                     | State    | FinalStatus |
|--------------------------------|--------|------------|------------------|---------|--------------------------------|--------------------------------|----------|-------------|
| application_1495255119959_0003 | hadoop | word count | MAPREDUCE        | default | Sat May 20 14:56:38 +0900 2017 | Sat May 20 14:57:11 +0900 2017 | FINISHED | SUCCEEDED   |

Showing 1 to 1 of 1 entries



**hadoop Application**  
application\_1495255119959\_0003

Kill Application

|                             |                                |
|-----------------------------|--------------------------------|
| User:                       | hadoop                         |
| Name:                       | word count                     |
| Application Type:           | MAPREDUCE                      |
| Application Tags:           |                                |
| YarnApplicationState:       | FINISHED                       |
| Queue:                      | default                        |
| FinalStatus Reported by AM: | SUCCEEDED                      |
| Started:                    | Sat May 20 14:56:38 +0900 2017 |
| Elapsed:                    | 33sec                          |
| Tracking URL:               | History                        |
| Diagnostic:                 |                                |

## 과제

## □ 강의 시간의 실습 내용을 정리하여 제출

- 하둡 실행 및 동작 확인
- 앞의 wordcount 맵리듀스 프로그램을 임의의 데이터에 대해 실행
- 웹 접속도 포함

❑ Apache Hadoop

- <https://hadoop.apache.org/>

❑ Hadoop Architecture and Deployment

- <http://www.rosebt.com/blog/hadooparchitecture-and-deployment>

❑ How to install Hadoop on Ubuntu 18.04 Bionic Beaver Linux

- <https://linuxconfig.org/how-to-install-hadoop-on-ubuntu-18-04-bionic-beaver-linux>