IV-2. Hadoop Eco System 관리

- Author: JinKyoung Heo
- Issue Date: 2014.08.19
- Revision #: Rev 6
- Homepage: www.javaspecialist.co.kr
- Email: hjk7902@gmail.com

이 페이지는 여백 페이지입니다.

1 Ganglia

Objectives

- 하둡 클러스터 모니터링 도구에 대해서 알아봅니다.

이 페이지는 여백 페이지입니다.

Ganglia

- 클러스터의 사용 상태를 모니터해 주는 도구
- UC Berkerley의 Millennium Project에서 개발
  - http://www.millennium.berkeley.edu/
- 홈페이지
  - http://ganglia.sourceforge.net/
  - http://ganglia.info/
- 메모리, CPU, 디스크, 네트워크 사용량 뿐만 아니라 몇 가지 설치/설정만으로 Hadoop에서 dfs, mapred 와 관련된 200여개 이상의 Hadoop 성능지표를 제공

Ganglia(ganglion의 복수)는 클러스터의 사용 상태를 모니터해 주는 도구입니다.

UC Berkerley의 Millennium Project(http://www.millennium.berkeley.edu/)에서 개발되었으며, SourceForge.net (http://ganglia.sourceforge.net/)를 통해 소스코드가 공개되어 자유롭게 사용할 수 있는 소프트웨어입니다.
2001년에 최초 버전이 릴리즈 된 이후로 지속적으로 업그레이드 되고 있습니다.

Ganglia는 Cacti와 함께 Hadoop Cluster 모니터링에 매우 유용한 오픈 소스 모니터링 툴 중에 하나입니다.

Ganglia는 Cluster내 노드들로 부터 성능지표들을 모니터링하고 종합 관제할 수 있는 기능을 제공합니다. 즉, 메모리, CPU, 디스크, 네트워크 사용량 뿐만 아니라 몇 가지 설치/설정만으로 Hadoop에서 dfs, mapred 와 관련된 200여개 이상의 Hadoop 성능지표를 제공하여 줍니다.

용어

- 노드
  - 메타 노드
    - 모니터링 되는 서버들의 성능지표들을 모으는 노드
  - 모니터링 노드
    - 모니터링 대상이 되는 노드
- gmond (Ganglia Monitoring Daemon)
  - 시스템의 상태를 모니터링하기 위한 데몬으로 포트번호 8649번을 사용하며, 이 포트를 통해 접속하면 시스템의 상태를 XML형식으로 출력하여 준다.
  - 모니터링 대상이 되는 모든 서버에 설치
- gmetad (Ganglia Meta Daemon)
  - 모니터링 웹 UI 를 제공하는 데몬
  - Meta노드에 설치된다.
- gstat (Ganglia Cluster Status Tool)
  - 클러스터의 상태를 커멘드라인 상에서 보기 위한 도구

ganglia Monitor Daemon (gmond)

모니터링하기를 원하는 모든 노드에 설치되어야 합니다. 다른 호스트의 gmond 데몬에 멀티캐스트 메시지를 보내어 자신의 상태를 알리며, 다른 호스트의 정보를 수집하여 자신과 다른 노드의 시스템 상태를 XML형식으로 알려줍니다.

ganglia Meta Daemon (gmetad)

gmond는 같은 네트웍상의 호스트에게만 호스트정보를 전달하기 때문에 WAN(Wide Area Network)에서는 멀티캐스팅이 되지 않습니다. gmetad는 WAN에서 여러 gmond 데몬으로 부터 수집된 호스트 정보를 RRD(round-robin database)에 저장합니다. gmetad는 ganglia Web Interface가 설치될 웹서버에만 동작하면 됩니다.

ganglia Web Interface

웹 인터페이스 모듈은 PHP언어로 작성되어 있으며 PHP가 지원되는 웹서버상에서 동작합니다. gmetad에 의해 수집되어 RRD에 저장된 시스템 정보를 보여주는 역할을 합니다.

Ganglia 설치환경 및 다운로드

◈ 설치 환경
- OS : CentOS 6.4
- Hadoop 2.2.0
  - Namenode 1대
  - Secondary namenode 1대
  - Datanode 2대
- Ganglia 3.6.0
  - Ganglia node 1대

◈ 다운로드
- http://ganglia.sourceforge.net/ -> Downloads
- ganglia monitor core
  - ganglia-3.6.0.tar.gz
- ganglia Web
  - ganglia-web-3.5.12.tar.gz

설치할 환경의 하둡 클러스터 환경은 다음과 같습니다.

네임노드 (master) 1대

데이터노드 (slave1, slave2) 2대

보노네임노드 (backup) 1대

Ganglia 설치

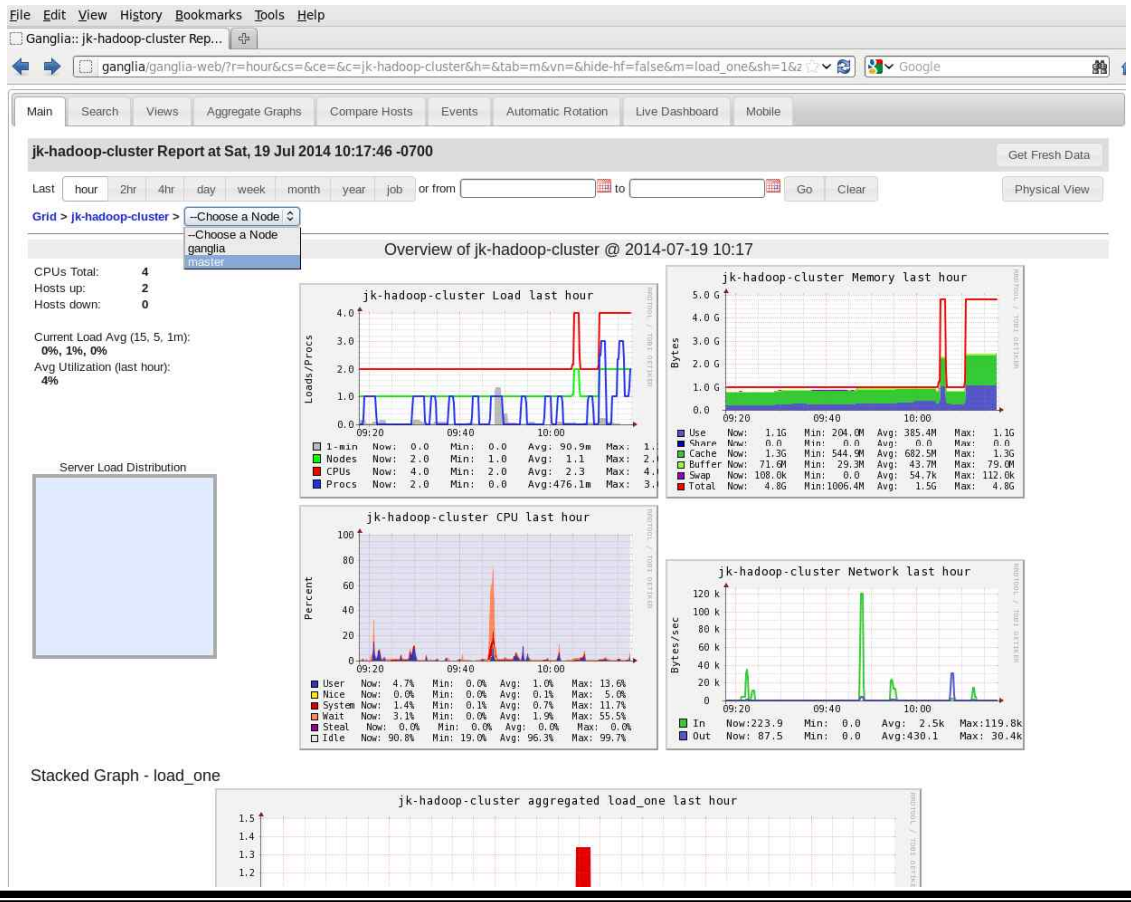
메타데몬노드(ganglia) : 메타 데몬(gmetad) 설치

하둡 클래스터(master, backup, slave1, slave2) : 모니터 데몬(gmond) 설치

Ganglia 노드를 별도로 설치하여 메타노드로 이용합니다.

Ganglia 설치

- ◈ 의존성 라이브러리 설치
- ◈ 메타 노드
  - ➢ ganglia 메타데몬 설치
    - ▪ configure, make, make install
    - ▪ 설정파일
  - ➢ ganglia web 설치
  - ➢ ganglia 데몬 실행
  - ➢ ganglia 데몬 설정파일
- ◈ 마스터 노드
  - ➢ hadoop 설정파일 수정
  - ➢ 설정파일을 데이터 노드에 배포
- ◈ 모니터 노드
  - ➢ ganglia 모니터데몬 설치
  - ➢ 모니터 노드는 ganglia와 master를 이용



Ganglia 설치 (메타노드)

1. 설치하기 전에 의존성 라이브러리들을 설치해야 합니다.(패키지들 root 계정으로 설치합니다.)

```
[root@ganglia ~]# yum install rrdtool-devel
[root@ganglia ~]# yum install apr-devel
[root@ganglia ~]# rpm -Uvh http://apt.sw.be/redhat/el6/en/i386/rpmforge/RPMS/rpmforge-release-0.5.3-1.el6.rf.i686.rpm
[root@ganglia ~]# yum install libconfuse-devel
[root@ganglia ~]# yum install expat-devel
[root@ganglia ~]# yum install pcre-devel
[root@ganglia ~]# yum install zlib-devel
```

libconfuse 패키지를 설치하기 위함

* PCRE(Perl Compatible Regular Expressions) 란 UTM이 트래픽을 처리하는 과정에서 패킷을 모니터링하며 실시간으로 트래픽을 분석하고 IP네트워크상에서 침입탐지의 패턴을 분석하는 일종의 패턴룰입니다.

2. 다운로드 받은 디렉토리에 압축을 풀어줍니다.

```
[root@ganglia Downloads]# tar -xvf /home/hadoop/Downloads/ganglia-3.6.0.tar.gz
```

3. 시스템과 컴파일 환경을 조사해서 Makefile을 만듭니다. 설치될 디렉토리는 /usr/local/ganglia 로 합니다.

```
[root@ganglia ganglia-3.6.0]# ./configure --prefix=/usr/local/ganglia --with-gmetad
...
config.status: executing libtool commands

Welcome to..
      ______                    __
     / ____/___ _____  ____ _/ (_)___ _
    / / __/ __ `/ __ ₩/ __ `/ / / __ `/
   / /_/ / /_/ / / / / /_/ / / / /_/ /
   ₩____/₩__,_/_/ /_/₩__, /_/_/₩__,_/
                     /____/

Copyright (c) 2005 University of California, Berkeley

Version: 3.6.0
Library: Release 3.6.0 0:0:0

Type "make" to compile.
```

4. 위와 같은 화면이 보이면 컴파일하고 설치합니다.

```
[root@ganglia ganglia-3.6.0]# make
[root@ganglia ganglia-3.6.0]# make install
```

Ganglia 설치 (메타노드)

5. gmetad.conf 파일에서 rrd 디렉토리 경로를 수정합니다.(129라인을 수정하면 됩니다.)

```
[root@ganglia ganglia-3.6.0]# vi /usr/local/ganglia/etc/gmetad.conf
rrd_rootdir "/usr/local/ganglia/rrds"
data_source "jk-hadoop-cluster" 5 ganglia master backup
```

- data_source 를 지정하고 5는 Interval 5초, 그 다음은 gmetad, gmond 가 설치된 Server side 의 서버(ganglia)를 첫 번째로 기술해야 한다.
- master, slave1, slave2, backup은 클러스터 호스트를 기술한다.(host명은 /etc/hosts 참조)
- 나머지는 default 설정으로 해도 무관함

6. gmond 설정파일 생성

```
[root@ganglia ganglia-3.6.0]# cd /usr/local/ganglia/sbin/
[root@ganglia sbin]# ./gmond --default_config > /usr/local/ganglia/etc/gmond.conf
```

7. gmond.conf 파일 수정

```
[root@ganglia sbin]# cd /usr/local/ganglia/etc
[root@ganglia etc]# vi /usr/local/ganglia/etc/gmond.conf
globals {
  daemonize = yes
  setuid = yes
  user = onbody
  debug_level = 0
  max_udp_msg_len = 1472
  mute = no
  deaf = no
  allow_extra_data = yes
  host_dmax = 86400 /*secs. Expires (removes from web interface) hosts in 1 day */
  host_tmax = 20 /*secs */
  cleanup_threshold = 300 /*secs */
  gexec = no
  send_metadata_interval = 0 /*secs */
}

cluster {
  name = "jk-hadoop-cluster"  /* gmetad.conf 에서 지정한 data_source와 일치 해야 함*/
  owner = "heojk"     /*owner, latlog, url 은 기입 해도 되고 안 해도 무관*/
  latlong = "unspecified"
  url = "http://javaspecialist.co.kr"
}
```

Ganglia 설치 (메타노드)

```
udp_send_channel {
  # mcast_join = 239.2.11.71
  host = ganglia  #send channel에 ganglia 메타데몬이 설치된 hostname을 기입한다.
  port = 8649
  ttl = 1
}
udp_recv_channel {
  # mcast_join = 239.2.11.71
  port = 8649
  # bind = 239.2.11.71
  # retry_bind = true
  # Size of the UDP buffer. If you are handling lots of metrics you really
  # should bump it up to e.g. 10MB or even higher.
  # buffer = 10485760
}
```

8. rrd 디렉토리를 생성한 다음 소유주를 변경하고 777권한을 부여해야 합니다.

```
[root@ganglia ganglia]# mkdir -p /usr/local/ganglia/rrds/
[root@ganglia Downloads]# chown nobody:nobody /usr/local/ganglia/rrds/
[root@ganglia Downloads]# chmpd 777 /usr/local/ganglia/rrds/
```

9. ganglia 소스 디렉토리에서 데몬을 시작시키기 위한 init 파일들을 /etc/init.d/ 에 복사합니다.

```
[root@slave1 ganglia]# cd /home/hadoop/Downloads/ganglia-3.6.0/
[root@ganglia ganglia-3.6.0]# cp ./gmetad/gmetad.init /etc/init.d/gmetad
[root@ganglia ganglia-3.6.0]# cp ./gmond/gmond.init /etc/init.d/gmond
```

10. ganglia 설치 디렉토리 아래의 sbin 디렉토리의 실행파일들을 /usr/sbin/에 복사합니다.

```
[root@ganglia ganglia-3.6.0]# cp /usr/local/ganglia/sbin/* /usr/sbin/
```

11. ganglia web을 사용하기 위해 httpd를 설치합니다.

```
[root@ganglia ganglia]# cd /home/hadoop/Downloads/
[root@ganglia Downloads]# yum install httpd php
```

12. ganglia web 압축파일을 풀어서 아파치 웹서버 홈디렉토리(/var/www/html/)에 복사 합니다.

```
[root@ganglia Downloads]# tar xf ganglia-web-3.5.12.tar.gz
[root@ganglia Downloads]# mv ganglia-web-3.5.12 /var/www/html/ganglia-web/
[root@ganglia Downloads]# cp -rp /var/www/html/ganglia-web/conf_default.php
    /var/www/html/ganglia-web/conf.php";
```

Ganglia 설치 (메타노드)

13. ganglia web 설정파일을 수정합니다.

```
[root@ganglia Downloads]# vi /var/www/html/ganglia-web/conf.php
gweb_confdir(13라인) 수정 → $conf['gweb_confdir'] = "/var/www/html/ganglia-web";
gmetad_root(387라인) 수정 → $conf['gmetad_root'] = "/usr/local/ganglia
```

14. compiled와 cache 디렉토리 생성 및 권한 변경

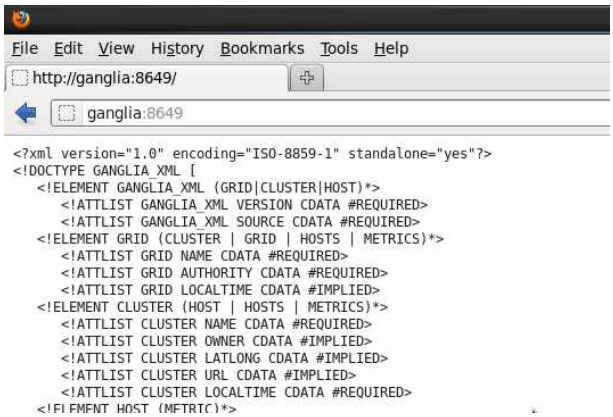
```
[root@ganglia Downloads]# mkdir -p /var/www/html/ganglia-web/dwoo/compiled
[root@ganglia Downloads]# mkdir -p /var/www/html/ganglia-web/dwoo/cache
[root@ganglia Downloads]# chmod 777 /var/www/html/ganglia-web/dwoo/compiled/
[root@ganglia Downloads]# chmod 777 /var/www/html/ganglia-web/dwoo/cache/
```

15. gmond와 gmetad 실행

```
[root@ganglia ~]# service gmetad start
[root@ganglia ~]# service gmond start
```

부팅시 자동 실행되게 하기 위해서는...
chkconfig gmond on
chkconfig gmetad on

16. 브라우저에서 테스트



17. ganglia-web 홈디렉토리 소유주 및 권한 변경 후 httpd 실행

```
[root@ganglia ~]# cd /var/www/html/
[root@ganglia html]# chown -R apache:apache ganglia-web/
[root@ganglia html]# chmod 755 ganglia-web/
[root@ganglia html]# service httpd start
```

18. /etc/selinux/config 파일에서 SELINUX=enforcing을 SELINUX=disabled로 변경합니다.

```
[root@ganglia html]# vi /etc/selinux/config
SELINUX=disabled
[root@ganglia html]# setenforce 0
```

Ganglia 설치 (메타노드)

19. 네임노드(master)에 hadoop 계정으로 접속합니다. 그리고

```
[hadoop@master ~]$ cd /usr/local/hadoop-2.2.0/etc/hadoop/
[hadoop@master hadoop]$ vi hadoop-metrics2.properties
*.sink.file.class=org.apache.hadoop.metrics2.sink.FileSink
#*.sink.ganglia.class=org.apache.hadoop.metrics2.sink.ganglia.GangliaSink31
#*.sink.ganglia.period=10

Namenode.sink.ganglia.servers=ganglia:8649
Datanode.sink.ganglia.servers=ganglia:8649
Jobtracker.sink.ganglia.servers=ganglia:8649
Tasktracker.sink.ganglia.servers=ganglia:8649
Maptask.sink.ganglia.servers=ganglia:8649
Reducetask.sink.ganglia.servers=ganglia:8649
```

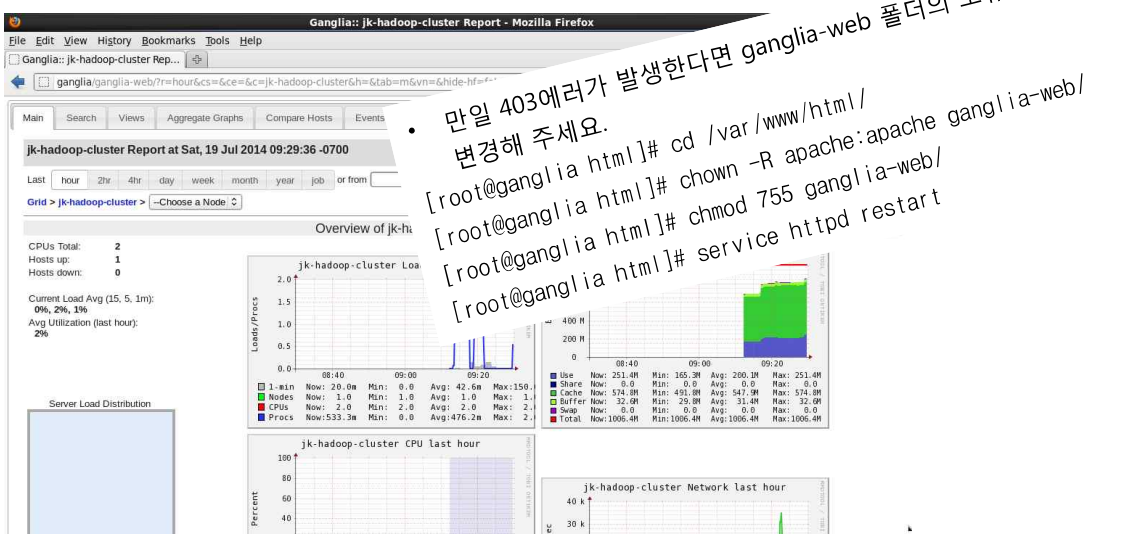
20. 위의 설정파일을 모든 데이터 노드들에 싱크 처리합니다.

```
[hadoop@master hadoop]$ rsync -av ./hadoop-metrics2.properties
    hadoop@slave1:/usr/local/hadoop-2.2.0/etc/hadoop/
```

21. 하둡 클러스터를 시작시킵니다.

```
[hadoop@master hadoop]$ cd /usr/local/hadoop-2.2.0/sbin/
[hadoop@master sbin]$ ./start-all.sh
```

22. 브라우저에서 http://ganglia/ganglia-web 으로 접속합니다.



- 만일 403에러가 발생한다면 ganglia-web 폴더의 소유주와 권한을 변경해 주세요.

```
[root@ganglia html]# cd /var/www/html/
[root@ganglia html]# chown -R apache:apache ganglia-web/
[root@ganglia html]# chmod 755 ganglia-web/
[root@ganglia html]# service httpd restart
```

Ganglia 설치 (모니터노드)

1. 컴파일 및 설치

```
[root@master ganglia-3.6.0]# ./configure --prefix=/usr/local/ganglia
[root@master ganglia-3.6.0]# make
[root@master ganglia-3.6.0]# make install
```

2. 모니터 데몬 설정파일 생성 및 수정

```
[root@master sbin]# mkdir /usr/local/ganglia/etc
[root@master sbin]# cd /usr/local/ganglia/sbin/
[root@master sbin]# ./gmond --default_config > /usr/local/ganglia/etc/gmond.conf
[root@master etc]# vi /usr/local/ganglia/etc/gmond.conf
cluster {
  name = "jk-hadoop-cluster"
  owner = "unspecified"
  latlong = "unspecified"
  url = "http://javaspecialist.co.kr"
}
udp_send_channel {
  host = ganglia
  port = 8649
  ttl = 1
}
udp_recv_channel {
  port = 8649
}
tcp_accept_channel {
  port = 8649
}
```

3. 서비스 자동 실행을 위한 설정

```
[root@master ganglia]# cp /usr/local/ganglia/sbin/* /usr/sbin/
[root@master ganglia]# cd /home/hadoop/Downloads/ganglia-3.6.0
[root@master ganglia-3.6.0]# cp ./gmond/gmond.init /etc/init.d/gmond
[root@master ~]# chkconfig gmond on
[root@master ~]# service gmond start
```

4. ganglia 호스트에서 gmetad 재시작, httpd 재시작

```
[root@ganglia html]# service gmetad restart
[root@ganglia html]# service httpd restart
```

5. 브라우저에서 확인(http://ganglia/ganglia-web) → Node에 master 추가되어 있어야 함.

2 Zookeeper

Objectives

- 하둡 클러스터 관리 도구 Zookeeper에 대해서 알아봅니다.

이 페이지는 여백 페이지입니다.

ZooKeeper 소개

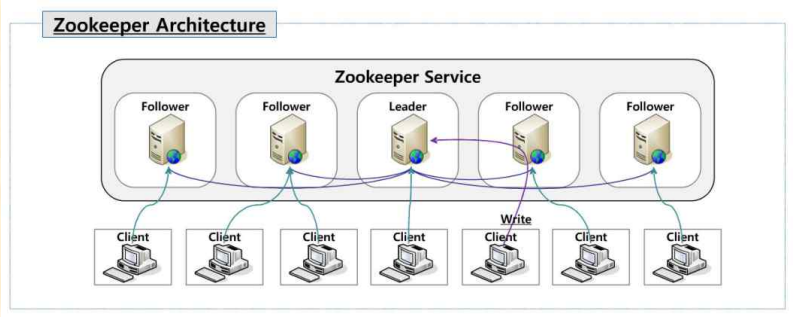
- Zookeeper는 고가용성 분산 코디네이션 시스템
- Zookeeper의 주요 용도
  - 분산 락킹(distributed locking)
  - 분산 시스템의 통합 설정 관리
  - 네이밍 서비스
  - 분산 시스템의 단일 스퀀스
  - Active/Stanby 관리, 클러스터의 멤버쉽 관리
  - 분산 시스템 각 머신의 Life 상태 / 자원 모니터링
- Zookeeper 특징
  - 고가용성, 확장성, 분산성, 공유 설정
  - 합의체, 그룹 멤버쉽, 리더 선출
  - 네이밍/코디네이션

ZooKeeper 개발 동기

- 기존 코디네이터의 한계
  - Active/Stanby 구성에서 부하가 단일 Active에 집중
  - 특정 시스템에 종속적인 코디네이터
- 분산환경에서 코디네이터의 필요성
  - 어플리케이션에서 고가용성 락 사용
  - 신뢰성을 가진 스퀀스 사용
  - 어플리케이션에서 쉽게 사용할 수 있는 코디네이터
- 새로운 코디네이터의 요구사항
  - 간단하고 견고, 고성능
  - 기존과 유사한 모델과 인터페이스
  - 부하의 분산, 빠른 반응 속도
  - 이벤트 대기 시 저부하
- 하둡과 별도로 개발됐지만 하둡의 핵심 구성원으로 자리잡음

ZooKeeper 구조

- Leader/Follower 구조
- 모든 서버에서 Read/Write 가능
- 리더가 받아서 다른 Follower에 전달



- 모든 Zookeeper 서버는 데이터의 복제본을 저장
- 리더는 시작 시 리더선출 알고리즘에 의해서 자동 선정(리더 장애시에도)
- 이벤트는 리더로 전송되고 모든 서버에 저장될 때 결과는 리턴됨

ZooKeeper Data Model

- 인메모리 데이터베이스
  - 내구성을 위해 변경내역은 로그로 저장
  - 클라이언트는 파일시스템처럼 사용 가능
- 데이터는 계층화된 패스노드
  - znode
  - 디렉토리/파일 구조 아님(데이터를 가지는 디렉토리)
  - 데이터는 key-value로 저장
  - 데이터는 최대 1M 제한
- 임시 노드
  - 세션이 삭제되면 임시노드도 삭제
- 시퀀스 노드
- 고속 처리
  - 초당 50,000번 변경
  - 초당 200,000번 읽기

Zookeeper Operation / API

- 9개의 기본 Operation
- 각 노드는 ACL을 가짐
- ACL:Create/Read/Write/Delete/Admin

- Zookeeper API
  - String create(path, data, acl, flags)
  - void delete(path, expectedVersion)
  - Stat setData(path,data,expectedVersion)
  - (data, Stat) getData(path, watch)
  - Stat exists(path, watch)
  - String[] getChildren(path, watch)
  - void sync(path)

http://zookeeper.apache.org/doc/r3.4.6/api/org/apache/zookeeper/ZooKeeper.html

Operation	ACL	Description
create	Create	Creates a znode. (parent must already exist)
delete	Delete	Deletes a znode (must not have any children)
exists	Read	Tests whether a znode exists and retrieves its metadata
getACL, setACL	Admin	Gets/Sets the ACL(permissions) for a znode
getChildren	Read	Gets a list of the children of a znode
getData, setData	Read/Write	Gets/Sets the data associated with a znode
sync	-	Synchronizes a client's view of a znode

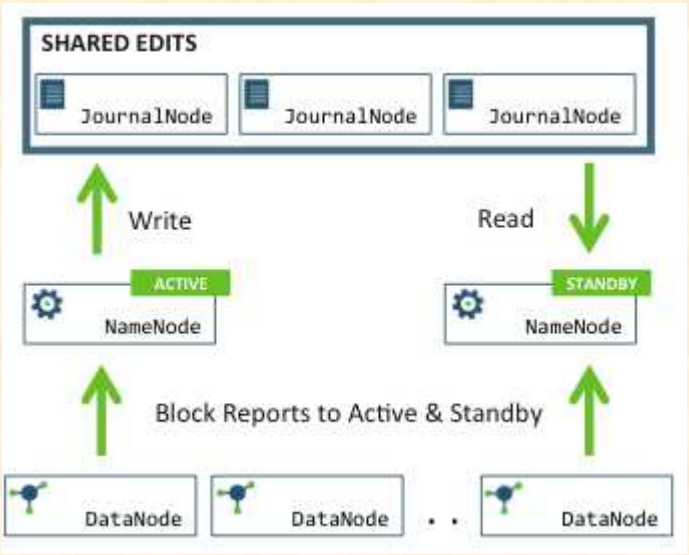
Watch

- Zookeeper는 Watch 라는 개념을 지원
- 클라이언트는 특정 znode에 Watch를 설정
  - znode에 변경이 발생하면 클라이언트에 통보
  - 통보 후 에는 Watch는 삭제됨(주의:필요한 경우 재설정)
- 서버가 변경 시 통보하기 때문에 부하가 적음
  - 클라이언트가 서버에 변경이 되었는지 모니터링 하지 않음
- 장애처리
  - 1.연결된 Zookeeper서버 장애
    - 다른 Zookeeper 서버로 대체
  - 2.클라이언트와 Zookeeper와의 연결이 단절될 경우?
    - local notification, 클라이언트에서 예외처리

ZooKeeper의 보장

- 순차적 일관성
  : 순서대로 처리됨
- 원자성
  : 성공과 실패 중 하나.
- 단일 시스템 이미지
  : 클라이언트는 연결된 서버를 통해 동일한 정보를 봄
- 신뢰성
  : 변경이 발생하면 모든 클라이언트가 변경내역을
  갱신할때 까지 기존정보는 유지됨
- 적시성
  : 특정 시간내에 변경사항이 모두 통보됨

HDFS High Availability Using the Quorum Journal Manager



Hardware Resources

- NameNode
  - Active and Standby NameNodes
  - 동일한 하드웨어 사양을 권장
- JournalNode
  - 상대적으로 가벼운 데몬
  - 다른 하둡 클러스터(NN, RM등)와 같은 위치에 배치하도록 함
  - 홀수로 구성, 최소 3대(3, 5, 7 등)
- Zookeeper
  - 자동화된 페일오버 기능을 위함
  - 하둡 클러스터와 함께 배치

- HA 클러스터에서 대기 네임 노드는 네임 스페이스 상태의 체크 포인트를 수행한다. 따라서, HA 클러스터의 보조 네임 노드, CheckpointNode 또는 BackupNode를 배포하지 않는다.

Zookeeper 설치

- http://zookeeper.apache.org/releases.html#download 에서 다운로드
- 주키퍼를 설치할 서버에는 JDK가 설치되어 있어야 함.
- zoo.cfg 설정 추가
  - tickTime=2000 #heartbeats 용, 세션 타임 관련 있음
  - dataDir=/home/hadoop/zookeeper/data #데이터 저장 디렉토리 지정
  - maxClientCnxns=0 #클라이언트 제한을 unlimit로 설정
  - server.1=node1:2888:3888
    - #Zookeeper 서버, 포트는 Zookeeper가 동작하기 위해서 필요한 포트, 앞 포트는 서버 간의 통신, 뒤의 포트는 Leader 선출에 사용
- dataDir 경로에 myid 파일 생성(각 서버에 자신의 ID(숫자)를 적어준다)
  - 1
  - node1 #hostname
- Zookeeper 서버 실행
  - ./bin/zkServer.sh start

1. 다운로드 받은 파일 압축 풀어 설치하기

```
[hadoop@node1 ~]$ su
Password:
[root@node1 Desktop]# cd /usr/local/
[root@node1 local]# tar -xf /home/hadoop/Downloads/zookeeper-3.4.6.tar.gz
[root@node1 local]# chown -R hadoop:hadoop zookeeper-3.4.6/
[root@node1 local]# ls -l
total 48
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 bin
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 etc
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 games
drwxr-xr-x. 11 hadoop hadoop 4096 Aug 17 21:21 hadoop-2.2.0
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 include
drwxr-xr-x.  8 hadoop hadoop 4096 Dec 18  2013 jdk1.7.0_51
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 lib
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 libexec
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 sbin
drwxr-xr-x.  5 root   root   4096 Feb 18 13:06 share
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 src
drwxr-xr-x. 10 hadoop hadoop 4096 Feb 20 02:58 zookeeper-3.4.6
[root@node1 local]# exit
exit
[hadoop@node1 ~]$ cd
```

Zookeeper 설치

2. 환경변수 설정

```
[hadoop@node1 ~]$ vi .bash_profile
…
export ZOOKEEPER_HOME=/usr/local/zookeeper-3.4.6
export PATH=$PATH: … :$ZOOKEEPER_HOME/bin
```

node2와 node3 서버에도 환경별수 설정해 주세요.

3. Zookeeper 설정파일 작성

```
[hadoop@node1 ~]$ cd /usr/local/zookeeper-3.4.6/conf/
[hadoop@node1 conf]$ cp zoo_sample.cfg zoo.conf
[hadoop@node1 conf]$ vi zoo.conf
# 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/hadoop/zookeeper/data
# the port at which the clients will connect
clientPort=2181
maxClientCnxns=0
server.1=node1:2888:3888
server.2=node2:2888:3888
server.3=node3:2888:3888
```

4. node2, node3 서버에 Zookeeper 복사

위의 내용을 node1, node2, node3 서버에 모두 설치해 줍니다.

만일 node1에 설치했다면 rsync 명령으로 다른 노드에 동기화 시켜줄 수 있습니다.

```
[root@node1 ~]# cd /usr/local
[root@node1 local]# rsync -av ./zookeeper-3.4.6/ root@node2:/usr/local/zookeeper-3.4.6/
[root@node1 local]# rsync -av ./zookeeper-3.4.6/ root@node3:/usr/local/zookeeper-3.4.6/
```

Zookeeper 설치

5. dataDir 디렉토리에 myid 파일 생성

```
[hadoop@node1 ~]$ mkdir /home/hadoop/zookeeper/data
[hadoop@node1 ~]$ vi myid
1
node1

[hadoop@node2 ~]$ mkdir /home/hadoop/zookeeper/data
[hadoop@node2 ~]$ vi myid
2
node2

[hadoop@node3 ~]$ mkdir /home/hadoop/zookeeper/data
[hadoop@node3 ~]$ vi myid
3
node3
```

6. Zookeeper 서버 실행

```
[hadoop@node1 zookeeper-3.4.6]$ ./bin/zkServer.sh start
JMX enabled by default
Using config: /usr/local/zookeeper-3.4.6/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
```

7. Zookeeper 서버 실행 확인

```
[hadoop@node1 zookeeper-3.4.6]$ jps
2851 QuorumPeerMain
2878 Jps
```

- 프로세스가 보이지 않을 때에는 실행한 디렉토리에 생성된 zookeeper.out 파일에서 에러를 확인해 보세요.

8. Zookeeper 클라이언트 실행

```
[hadoop@node1 zookeeper-3.4.6]$ ./bin/zkCli.sh
Connecting to localhost:2181
2014-08-18 06:46:44,754 [myid:] - INFO  [main:Environment@100] - Client
environment:zookeeper.version=3.4.6-1569965, built on 02/20/2014 09:09 GMT
2014-08-18 06:46:44,757 [myid:] - INFO  [main:Environment@100] - Client
environment:host.name=master
2014-08-18 06:46:44,757 [myid:] - INFO  [main:Environment@100] - Client
environment:java.version=1.7.0_51
2014-08-18 06:46:44,759 [myid:] - INFO  [main:Environment@100] - Client
environment:java.vendor=Oracle Corporation
```

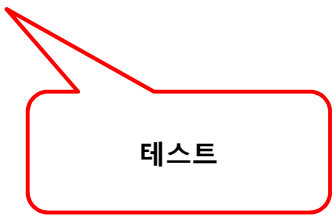
Zookeeper 설치

```
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client
environment:java.home=/usr/local/jdk1.7.0_51/jre
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client
environment:java.class.path=/usr/local/zookeeper-3.4.6/bin/../build/classes:/usr/local/zookeeper-
3.4.6/bin/../build/lib/*.jar:/usr/local/zookeeper-3.4.6/bin/../lib/slf4j-log4j12-
1.6.1.jar:/usr/local/zookeeper-3.4.6/bin/../lib/slf4j-api-1.6.1.jar:/usr/local/zookeeper-
3.4.6/bin/../lib/netty-3.7.0.Final.jar:/usr/local/zookeeper-3.4.6/bin/../lib/log4j-
1.2.16.jar:/usr/local/zookeeper-3.4.6/bin/../lib/jline-0.9.94.jar:/usr/local/zookeeper-
3.4.6/bin/../zookeeper-3.4.6.jar:/usr/local/zookeeper-3.4.6/bin/../src/java/lib/*.jar:/usr/local/zookeeper-
3.4.6/bin/../conf:
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client
environment:java.library.path=/usr/java/packages/lib/i386:/lib:/usr/lib
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client environment:java.io.tmpdir=/tmp
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client environment:java.compiler=<NA>
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client environment:os.name=Linux
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client environment:os.arch=i386
2014-08-18 06:46:44,760 [myid:] - INFO  [main:Environment@100] - Client environment:os.version=2.6.32-
358.el6.i686
2014-08-18 06:46:44,761 [myid:] - INFO  [main:Environment@100] - Client environment:user.name=hadoop
2014-08-18 06:46:44,761 [myid:] - INFO  [main:Environment@100] - Client environment:user.home=/home/hadoop
2014-08-18 06:46:44,761 [myid:] - INFO  [main:Environment@100] - Client environment:user.dir=/home/hadoop
2014-08-18 06:46:44,762 [myid:] - INFO  [main:ZooKeeper@438] - Initiating client connection,
connectString=localhost:2181 sessionTimeout=30000
watcher=org.apache.zookeeper.ZooKeeperMain$MyWatcher@183c7a0
Welcome to ZooKeeper!
JLine support is enabled
2014-08-18 06:46:44,840 [myid:] - INFO  [main-SendThread(127.0.0.1:2181):ClientCnxn$SendThread@975] -
Opening socket connection to server 127.0.0.1/127.0.0.1:2181. Will not attempt to authenticate using SASL
(unknown error)
2014-08-18 06:46:44,854 [myid:] - INFO  [main-SendThread(127.0.0.1:2181):ClientCnxn$SendThread@852] -
Socket connection established to 127.0.0.1/127.0.0.1:2181, initiating session
2014-08-18 06:46:44,882 [myid:] - INFO  [main-SendThread(127.0.0.1:2181):ClientCnxn$SendThread@1235] -
Session establishment complete on server 127.0.0.1/127.0.0.1:2181,  sessionid = 0x147e95885d70001,
negotiated timeout = 30000
[zk: localhost:2181(CONNECTED) 0]
WATCHER::

WatchedEvent state:SyncConnected type:None path:null
[Enter]
[zk: localhost:2181(CONNECTED) 0]
[zk: localhost:2181(CONNECTED) 0] help
ZooKeeper -server host:port cmd args
        connect host:port
        get path [watch]
        ls path [watch]
        set path data [version]
        rmr path
        delquota [-n|-b] path
        quit
        printwatches on|off
        create [-s] [-e] path data acl
```

에러 없이 여기까지 나오면 정상

Zookeeper 클라이언트 실행

```
        stat path [watch]
        close
        ls2 path [watch]
        history
        listquota path
        setAcl path acl
        getAcl path
        sync path
        redo cmdno
        addauth scheme auth
        delete path [version]
        setquota -n|-b val path
[zk: localhost:2181(CONNECTED) 1] get /zookeeper

cZxid = 0x0
ctime = Wed Dec 31 16:00:00 PST 1969
mZxid = 0x0
mtime = Wed Dec 31 16:00:00 PST 1969
pZxid = 0x0
cversion = -1
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 0
numChildren = 1
[zk: localhost:2181(CONNECTED) 2]
```



Configure NameNode HA Cluster

1. hdfs-site.xml 파일에 HA 설정 추가(Active Namenode와 Standby Namenode에 설정)

```
<property>
  <name>dfs.nameservices</name>
  <value>jk-hadoop-cluster</value>
</property>
<property>
  <name>dfs.ha.namenodes.jk-hadoop-cluster</name>
  <value>nn1,nn2</value>
</property>
<property>
  <name>dfs.namenode.rpc-address.jk-hadoop-cluster.nn1</name>
  <value>master:8020</value>
</property>
<property>
  <name>dfs.namenode.rpc-address.jk-hadoop-cluster.nn2</name>
  <value>master_standby:8020</value>
</property>
<property>
  <name>dfs.namenode.http-address.jk-hadoop-cluster.nn1</name>
  <value>master:50070</value>
</property>
<property>
  <name>dfs.namenode.http-address.jk-hadoop-cluster.nn2</name>
  <value>master_standby:50070</value>
</property>
<property>
  <name>dfs.namenode.shared.edits.dir</name>
  <value>qjournal://node1:8485;node2:8485;node3:8485/jk-hadoop-cluster</value>
</property>
<property>
  <name>dfs.client.failover.proxy.provider.jk-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>
```

```
[hadoop@master Desktop]$ cat /etc/hosts
192.168.224.152 master  backup  slave2  node1
192.168.224.149 slave1  node2
192.168.224.155 master2 node3
```

Configure NameNode HA Cluster

2. Active Namenode와 Standby Namenode에서 core-site.xml 파일에 디폴트 파일시스템 이름 변경 및 Quorum 서버 설정

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://jk-hadoop-cluster</value>
</property>
<property>
  <name>ha.zookeeper.quorum</name>
  <value>node1:2181,node2:2181,node3:2181 </value>
</property>
```

3. Journal Node 설정(안해도 된다?)

```
[hadoop@master hadoop-2.2.0]$ cd etc/hadoop/
[hadoop@master hadoop]$ vi journalnodes
node1
node2
node3
```

4. JournalNode들의 hdfs-site.xml 파일에 에디트 디렉토리 설정

```
<property>
  <name>dfs.journalnode.edits.dir </name>
  <value>/home/hadoop/journal/data</value>
</property>
```

journal의 edits 디렉토리는 절대경로로 입력해야 합니다.(URI 안됨)

5. hdfs-site.xml파일에 HA 자동화 설정

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

6. 네임노드 포맷

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

7. Standby 노드에 Active 노드의 메타 정보 복사

```
[hadoop@master ~]$ rsync -av ~/dfs/ hadoop@node3:./dfs/
```

8. Zookeeper Failover Cluster 포맷

```
[hadoop@master ~]$ ./hdfs zkfc -formatZK
```

Configure NameNode HA Cluster

8. 클러스터 시작

```
[hadoop@master sbin]$ ./start-dfs.sh
...
```

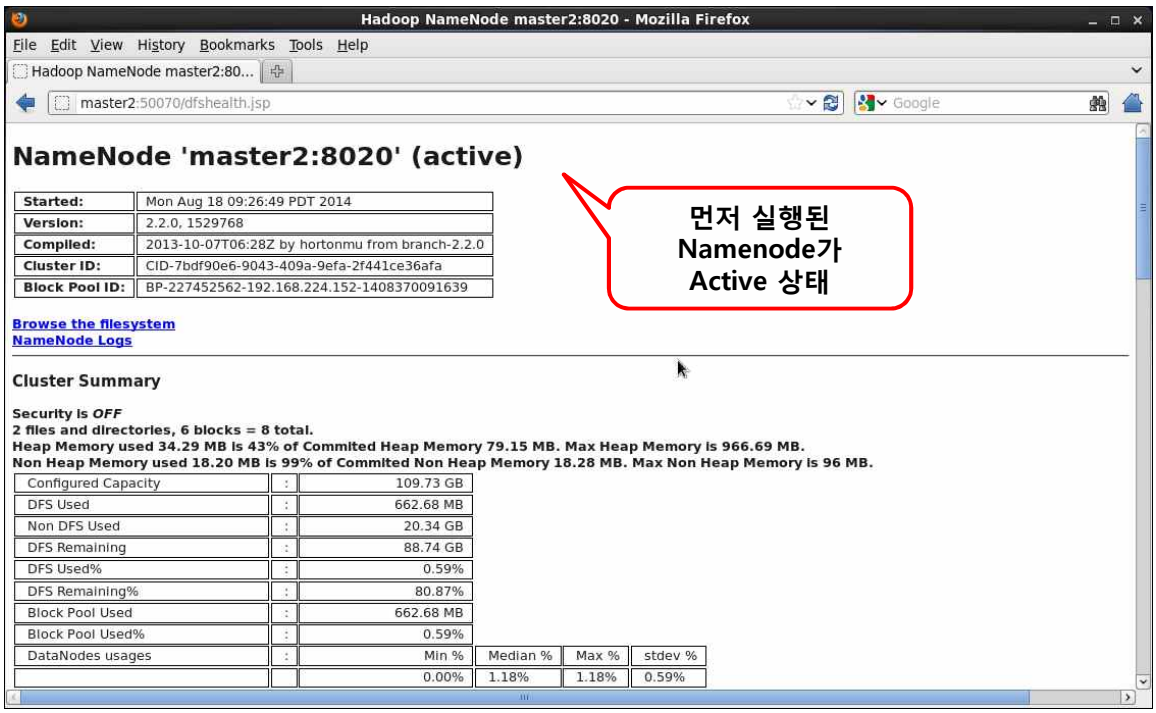
9. 프로세스 확인

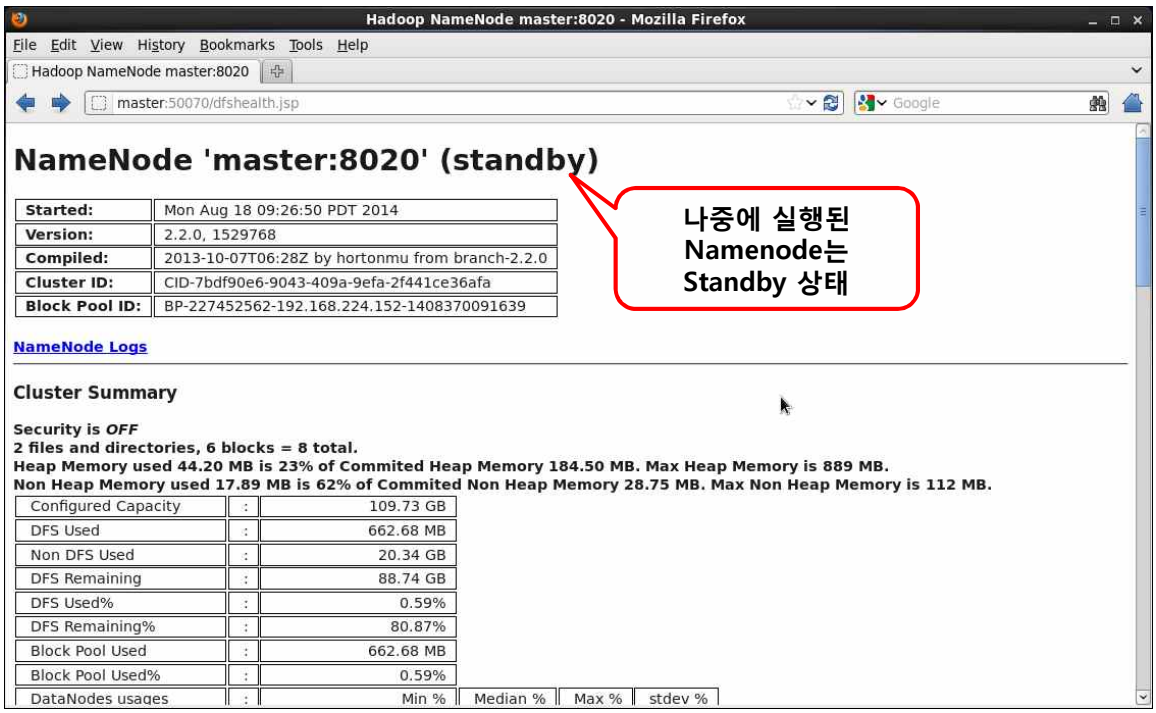
```
[hadoop@master sbin]$ jps
9041 NameNode
9564 Jps
9145 DataNode
9507 DFSZKFailoverController
2618 QuorumPeerMain
9330 JournalNode

[hadoop@master2 Desktop]$ jps
2899 NameNode
3132 Jps
2969 JournalNode
3072 DFSZKFailoverController
2743 QuorumPeerMain
```

* Zookeeper Failover Controller를 실행하는 방법은 다음과 같습니다.

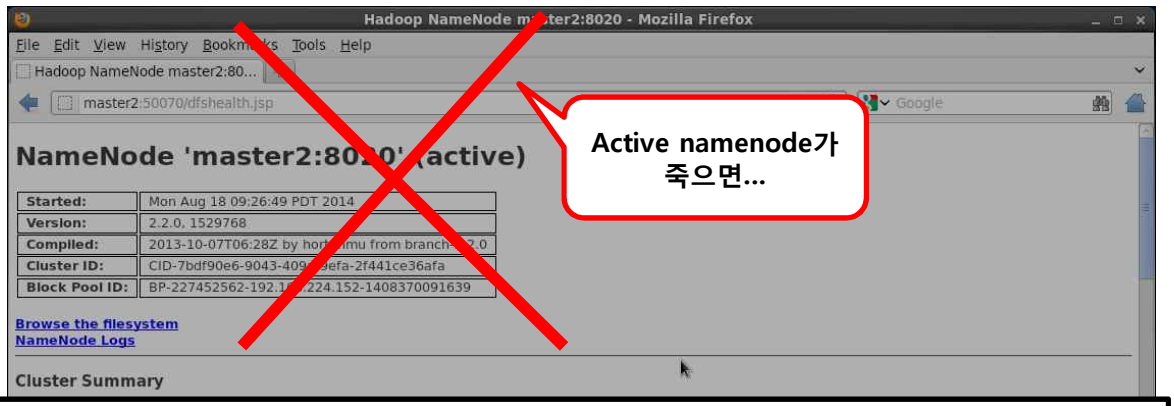
```
[hadoop@master sbin]$ ./hadoop-daemon.sh start zkfc
```

Configure NameNode HA Cluster



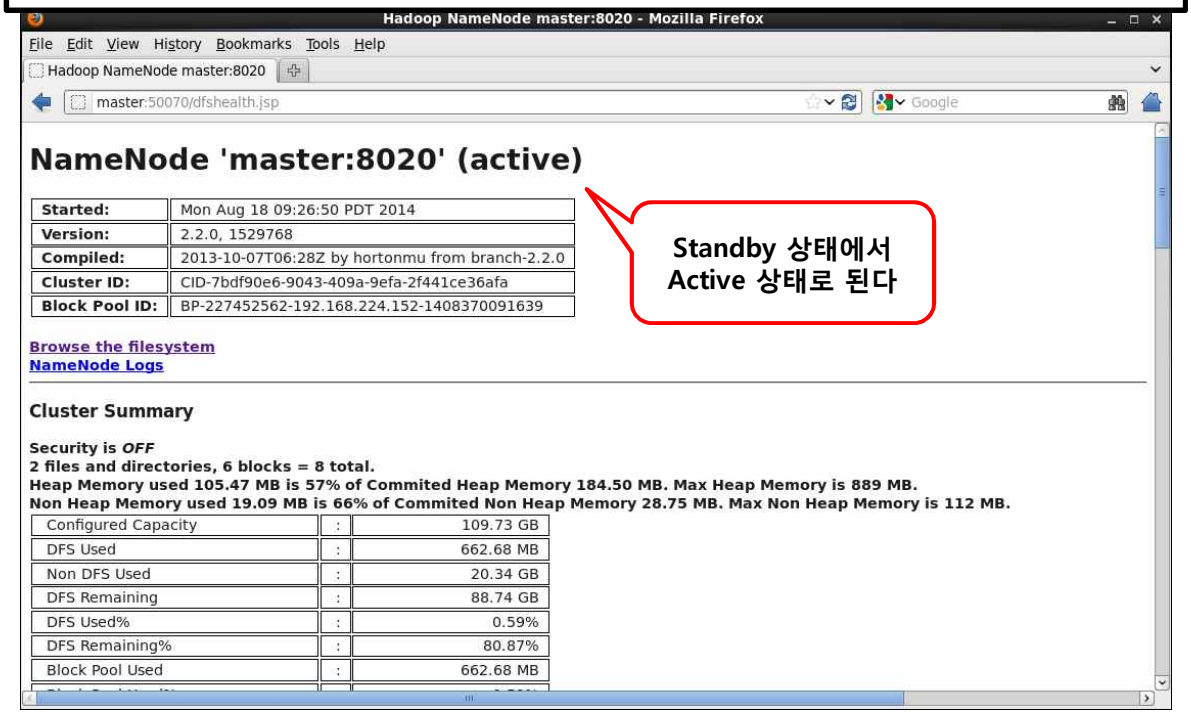


Configure NameNode HA Cluster




```
[hadoop@master2 sbin]$ ./hadoop-daemon.sh start zkfc
starting zkfc, logging to /usr/local/hadoop-2.2.0/logs/hadoop-hadoop-zkfc-master2.out
[hadoop@master2 sbin]$ ./hadoop-daemon.sh start namenode
starting namenode, logging to /usr/local/hadoop-2.2.0/logs/hadoop-hadoop-namenode-master2.out
[hadoop@master2 Desktop]$ zkServer.sh start
JMX enabled by default
Using config: /usr/local/zookeeper-3.4.6/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[hadoop@master2 Desktop]$ jps
2614 NameNode
2552 DFSZKFailoverController
3146 QuorumPeerMain
3188 Jps
```

다시 살릴 땐 zkfc와 namenode, zkServer 를 실행시킨다.



이 페이지는 여백 페이지입니다.