

# 목차

## 1. Service의 용도, 역할

Yarn, Map/Reduce, Zookeeper

Spark, Zeppelin

## 2. 시스템 구성도

VM, 네트워크, Service구성

## 3. Guide

HDP 관리페이지 접속방법

Hive 접속방법

Spark History 서버 접속 방법

Zeppelin 노트북 접속방법

## 4. QnA

QnA

## 1. Service의 용도, 역할

### Hadoop

대용량 처리를 위한 분산처리 프레임워크

- HDFS와 MapReduce Function 및 여러 에코시스템으로 구성
- 클러스터링기술을 통해 서버를 분산하여 I/O 병목현상을 줄임
- RDB 데이터 웨어하우스에 비해 경제적. Open Source

1) 확장이 쉬움

→ x86 서버를 증설하여 데이터노드를 설치 및 실행하면 스토리지 용량이 자동으로 늘어남

2) 메타 정보를 중앙에서 집중적으로 관리.

→ 데이터노드에 문제가 생겨도 파일 손실 위험이 없고 Redundancy를 피할 수 있음

3) 선형 디스크 입출력 분산

- 네임노드는 충분한 데이터노드가 있을시에 중복된 데이터노드 정보 반환X
- 모든 데이터노드의 localdis i/o 활용으로 Multi-tenancy를 확보

4) 선형 네트워크 부하 분산

→ 파일 크기가 크면 블록으로 쪼개어 저장. 한서버에 네트워크 트래픽을 오래 점유 하지 않음

5) 안정적 데이터 관리

- 파일의 복제본 수 지정가능
- 네트워크 Topology 기반으로 분산하여 안정성 확보

## 1. Service의 용도, 역할

### HDFS 구성요소

#### 1) 네임노드

- 필수 데몬 소프트웨어
- 시스템 전반상태 모니터링, 메타정보와 인덱스 정보를 갖고 있음
- 시스템 메모리, I/O에 대한 상태 모니터링

#### 2) 세컨더리 네임노드

- 보조 데몬 소프트웨어
- 정의된 값에 따라 주기적으로 네임노드의 파일 시스템 이미지를 스냅샷 후 생성 → 포인팅 서버

#### 3) 데이터노드

- 실질적인 데이터 입출력에 대한 처리 수행

#### 4) Job Tracker

- 사용자의 명령을 받아 데이터를 분석하기 위한 job 생성. 클러스터에는 하나의 잡트래커 데몬만 존재.

#### 5) Task Tracker

- 생성된 job 처리

## 1. Service의 용도, 역할

### ✓ MapReduce

분산처리 시스템

- 데이터 처리의 기본단위 : Mapper와 Reduce
- Map : 산재해 있는 데이터를 키와 벨류 형태로 연관성이 있는 데이터를 묶는 작업을 실행
- Reduce : 맵 작업결과에서 중복 데이터를 제거 한 후 원하는 데이터를 추출하는 작업 수행
- 많은 양의 데이터를 맵리듀스 형태로 작성 → 클러스터링 환경에서 효과적인 분산처리 가능

### ✓ MapReduceJob

- 블록 크기 설정 : 블록크기 크게 설정 Replication 위치 변경에 드는 비용, 작업량, 처리시간이 줄어든다
- 그렇다면 블록크기를 크게 하는 것이 좋은가 ? → No. 성능이 올라 갈 것 가지만 네임노드 힙 메모리에 영향을 주기 때문에 적절한 크기가 좋음 (기본 블록 크기 : 64MB 현 Setting : 32MB)
- Replication 숫자 : 클러스터의 크기에 따라 다름, 클러스터의 크기가 충분하면 복제 개수를 늘려 성능 향상 (현 Setting : 2)

## 1. Service의 용도, 역할

### 꼭 써야만 하는 걸까?

- 하둡은 뛰어난 병렬 처리 시스템
- MapReduce는 프로그램을 만드는데 시간이 많이 걸림
- 프로그램을 실행하기 위한 준비에도 시간이 필요
- 셔플링 단계가 존재하기 때문에 시간이 **더** 소요
- 그러나 데이터의 양이 많아지고 배치 작업으로 스케줄링 잡 관리시스템을 만들기 위해서 맵리듀스가 필요
- 데이터의 ETL 처리비용 또한 저렴.

## 1. Service의 용도, 역할

### 하둡 분산 파일 시스템을 구축하기 위한 시스템 요구 사항

#### 1. 리눅스 운영체제 → CentOS 이용

- 하둡은 리눅스 명령으로 시스템 정보를 수집하므로 리눅스 명령을 사용하는 운영체제가 필요함

#### 2. JDK 1.6.x 이상 버전 → 1.8.0\_112 이용

- 하둡은 자바 언어로 개발되었고 자바환경은 꼭 필요

#### 3. SSH 지원 → ssh private key인증

- 하둡은 ssh로 분산된 각 노드를 제어하기 때문에 ssh가 반드시 필요
- ssh 클라이언트는 하둡 실행시에는 필요 하지 않지만 하둡 설정 파일을 다른 서버에 배포할 때 유용
- key를 기반으로 통신 → 하둡을 실행하거나 중지할 때 매번 비밀번호를 입력하지 않아도 됨

## 1. Service의 용도, 역할

### YARN

- 자원 관리자 : 자원을 효율적으로 관리
- 리소스 매니저, 애플리케이션마스터, 노드 매니저로 구성
  - 1) 리소스 매니저(RM) : 클러스터 자원 중재, YARN에서 실행되는 분산 application관리, 스케줄링 역할을 하나 상태는 관리하지 않음.
  - 2) 애플리케이션 마스터 : 클러스터의 애플리케이션을 실행, 조정하는 데몬. 애플리케이션마다 애플리케이션 소유하는 AM이 존재. 자원교섭, 태스크 실행, 모니터링 역할
  - 3) 노드 매니저 : 개별 컴퓨팅 노드 관리. RM과 최신 상태 공유, 애플리케이션 컨테이너 생명주기 관리. 개별 컨테이너 자원 모니터링, 노드 상태 관리

## 1. Service의 용도, 역할



YARN



- YARN을 이용하기 위한 Service
- Name Node가 여러 개인 것 처럼 활용할 수 있다.



Map Reduce



- Map/Reduce Function을 이용하기 위한 프레임 워크



Zookeeper



- 환경에서 서버 간의 상호 조정이 필요한 다양한 서비스를 제공하는 시스템
- 하나의 서버에만 서비스가 집중되지 않게 분산해 동시에 처리
  - 처리한 결과를 다른 서버와도 동기화해서 데이터의 안정성을 보장
  - 운영서버가 문제가 발생해서 서비스를 제공할 수 없을 경우, 다른 대기 중인 서버를 운영서버로 바꿔서 서비스 중지 없이 제공



## 1. Service의 용도, 역할



HIVE



하이브는 HDFS에 저장한 데이터를 분석하기 위해서 맵리듀스를 사용하는 데이터 웨어하우스

업계 표준 SQL에 기반한 질의 언어를 제공한다. 코딩, 컴파일, 서버 및 과정을 없애고 HiveQL 문장만 조합하면 됨.

이런 방법을 통해 데이터 분석 결과물을 만드는 시간을 줄여줌



Spark



하둡과 유사한 클러스터 기반의 분산 처리 기능을 제공하는 오픈소스 프레임워크

처리결과를 항상 파일시스템에 유지하는 하둡과 달리 메모리에 저장하고 관리할 수 있는 기능을 제공함으로써 머신러닝 등 반복적인 데이터를 처리하는데 뛰어난 성능을 보임



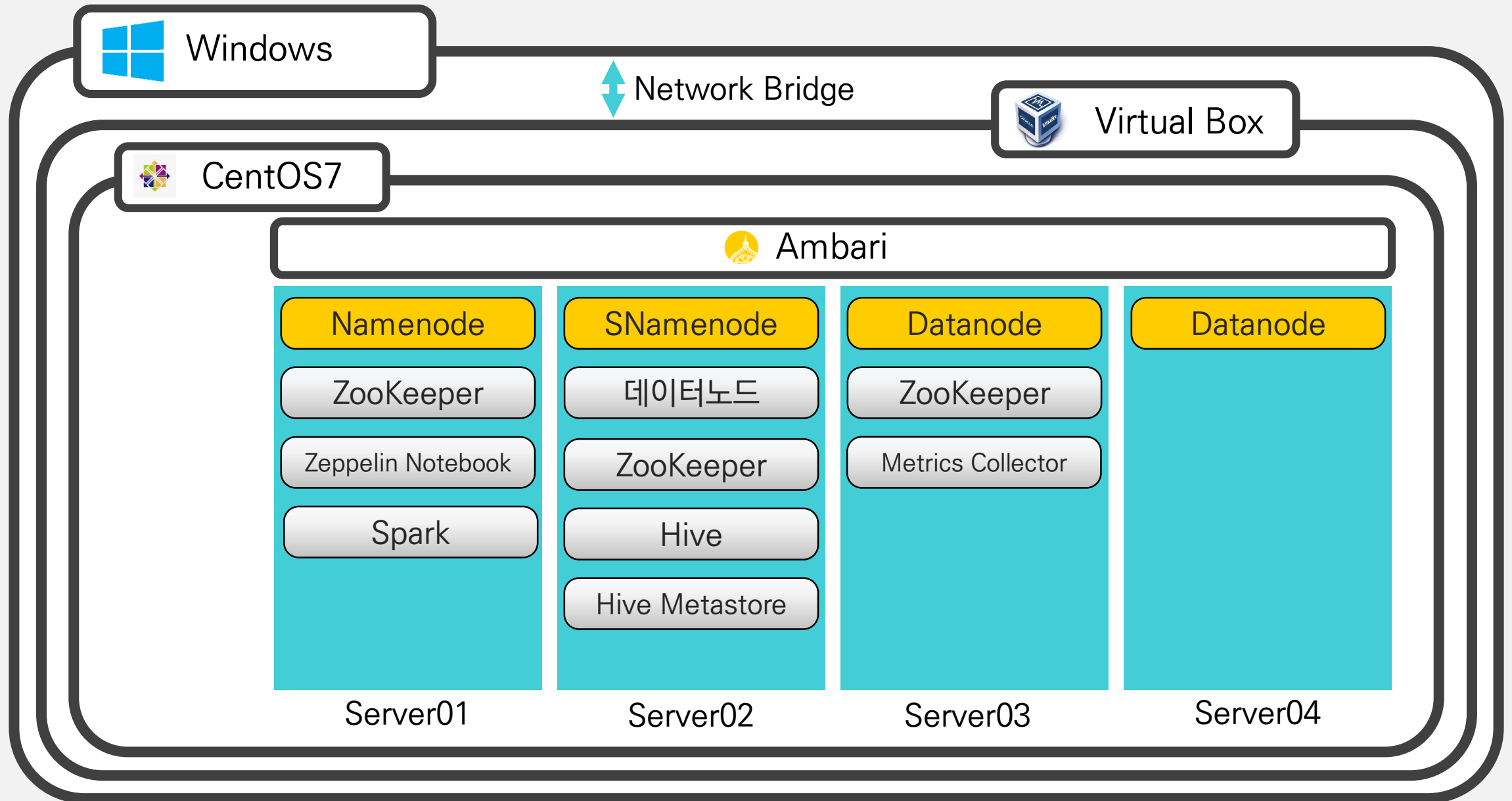
Zeppelin



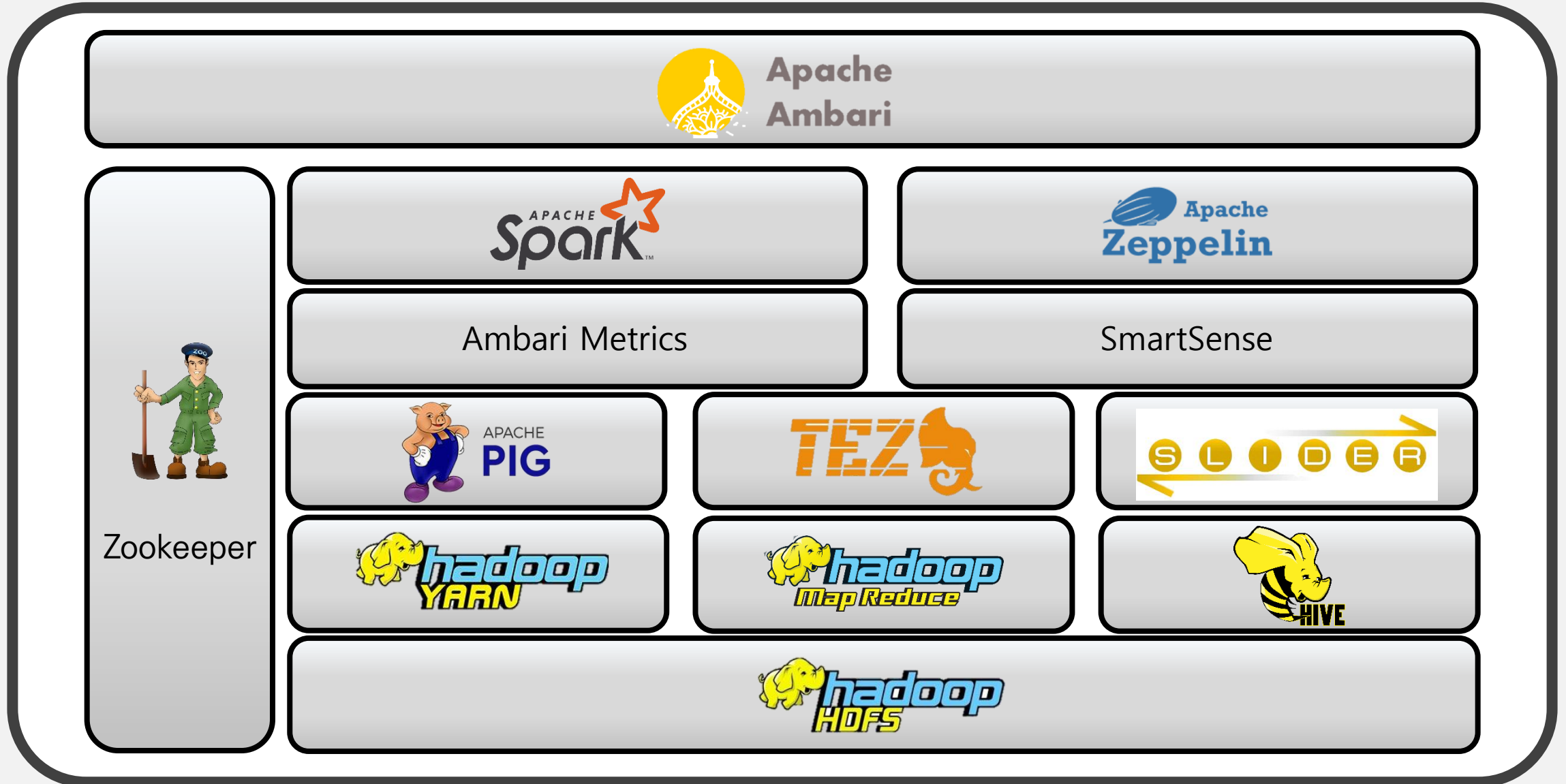
스파크를 기반으로 하여 하둡의 저장소에 있는 데이터를 직접 참조, 병렬처리가 가능한 분석 및 시각화 툴

웹 UI의 Notebook에서 스파크 또는 스파크 SQL을 작성해 하둡 클러스터에 작업을 요청하고, 처리결과를 다시 웹 UI에서 시각화해서 볼 수 있음

## 2. 시스템 구성도

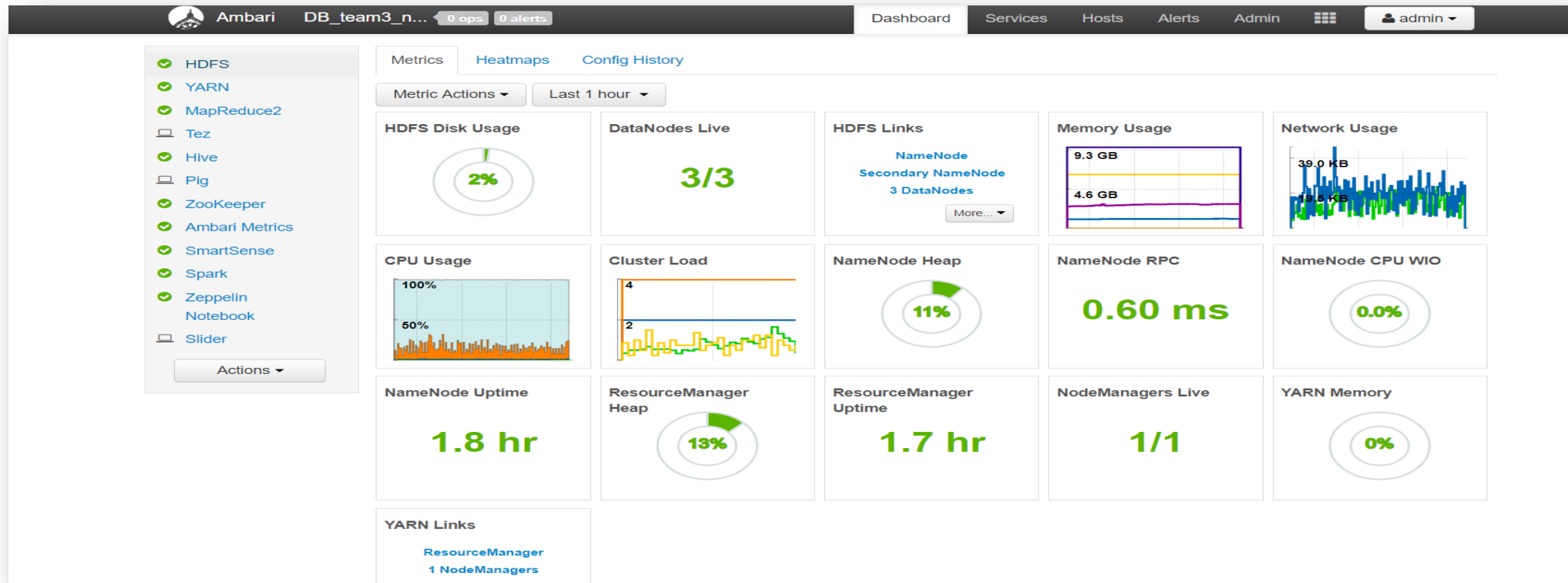


## 2. 시스템 구성도



### 3. HDP 관리 페이지 접속 방법

#### ✓ HDP 접속 방법



[root@server01 ~]# ambari-server start (네임노드)

[root@server01 ~]# ambari-agent start (네임노드, 데이터노드)

주소창에 네임노드 IP를 가지고 접속(디폴트 포트 8080)

EX) 192.168.113.221:8080

### 3. Hive 접속방법



Hive

```
[hdfs@server02 root]$ hive
log4j:WARN No such property [maxFileSize] in org.apache.log4j.DailyRollingFileAppender.

Logging initialized using configuration in file:/etc/hive/2.6.5.0-292/0/hive-log4j.properties
hive> █
```

```
[root@server02 ~]# su hdfs
```

```
[hdfs@server02 root]$ hive
```

```
Logging initialized using configuration in file:/etc/hive/2.6.5.0-292/0/hive-log4j.properties
```

```
hive>
```

### 3. Hive 접속방법



Hive

Hive Metastore

Hive Metastore host

server02.db.com

Hive Database

☐ New MySQL Database

☒ Existing MySQL / MariaDB Database

☐ Existing PostgreSQL Database

☐ Existing Oracle Database

To use MySQL with Hive, you must [download the MySQL Connector/J JDBC Driver from MySQL](#). Once downloaded to the Ambari Server host, run:  
**ambari-server setup --jdbc-db=mysql --jdbc-driver=/path/to/mysql/mysql-connector-java.jar**

Database Name

HIVE\_DATABASE

Database Username

hive

Database Password

.....

.....

JDBC Driver Class

com.mysql.jdbc.Driver

Database URL

jdbc:mysql://server02.db.com/HIVE\_DATABASE


Test Connection

Connection OK

### 3. Spark History 서버 접속 방법



#### Spark History server

 1.6.3

History Server

Event log directory: hdfs:///spark-history

Showing 1-2 of 2

App ID	App Name	Started	Completed	Duration	Spark User	Last Updated
<a href="#">local-1542881029349</a>	Spark shell	2018/11/22 19:03:45	2018/11/22 19:04:40	54 s	hdfs	2018/11/22 19:04:40
<a href="#">local-1542870288860</a>	Spark shell	2018/11/22 16:04:45	2018/11/22 16:06:32	1.8 min	hdfs	2018/11/22 16:06:32

[Show incomplete applications](#)

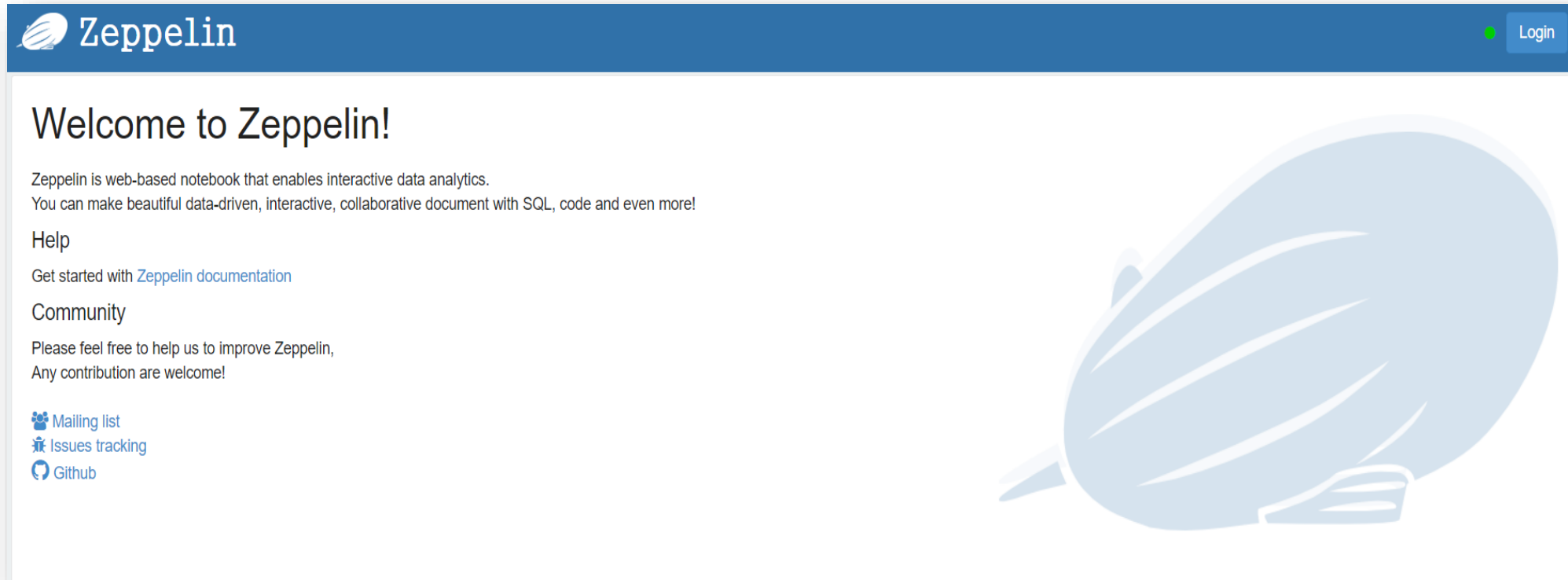
1

Spark History Server의 디폴트 포트인 18080을 이용하여 접속  
EX)192.168.113.221:18080

### 3. Zeppelin 노트북 접속 방법



Zeppelin



Zeppelin notebook의 디폴트 포트인 9995를 이용하여 접속  
EX) 192.168.113.221:9995