HDFS Router 기반 Multi Namenode 환경에서의 클러스터 성능 개선

(NC 소프트 인턴 김형근 fnfn9087@gmail.com)

(테스트 환경 : Hadoop 3.2.1, Hive 3.1.2, Zookeeper 3.5.8)

목차

1. 개요

선정 배경 : 클러스터 확장에 따른 Namenode 부담 증가. 따라서 부담을 감소시킬 필요성 존재

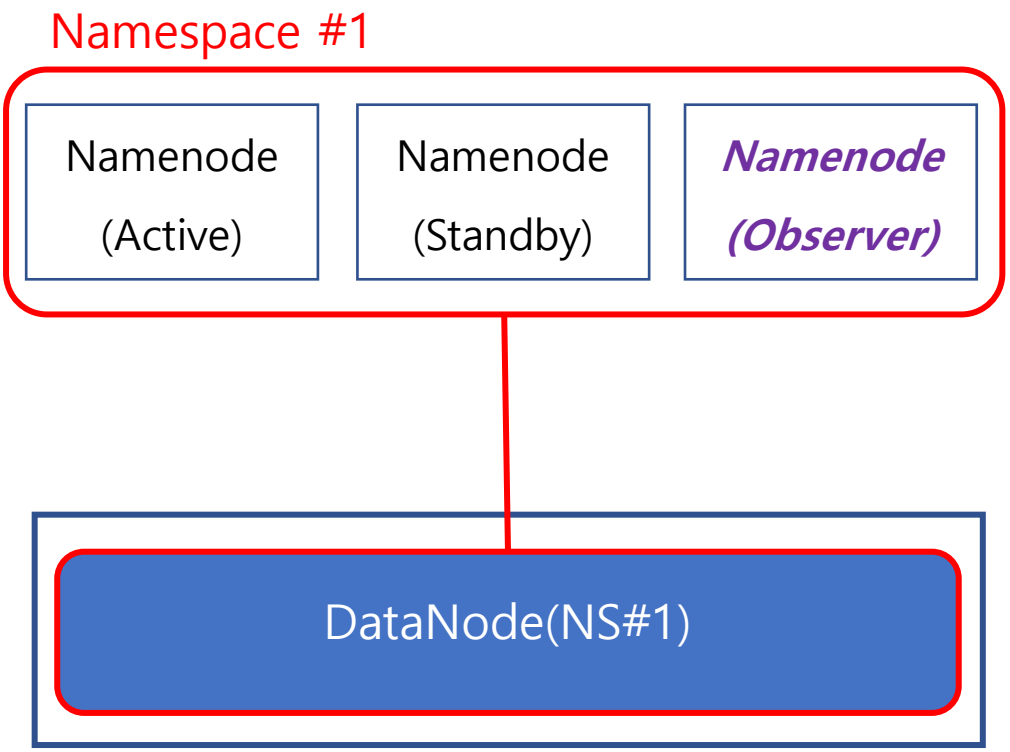
1) 안정성 검증 목표: *Observer Namenode 적용환경은 Legacy Active/Standby 환경보다 부하량 감소

2) 확장성 검증 목표: *HDFS Router *Multi Namespace 환경은 단일 Namespace 일 때 보다 성능이 향상

*[1]Observer Namenode : Active Node와 같이 Read 기능 제공. load balancing을 통한 부하 분산.
*[2]HDFS Router : 원하는 Namespace로 가기 위한 서비스. 원활한 Namespace 전환을 위해 구성.
*[3]Multi Namespace : 여러 대의 Namespace 환경을 Router 기반의 Federation으로 통합.

2. (1) 테스트 실행 – 안정성 검증

1) 클러스터 구조



Hadoop Service

1) Zookeeper (3.5.8) :
   - Namenode간 Active/Standby 상태 확인
   - Federation시 타 클러스터 존재를 기억

2) Hadoop (3.2.1) :
   - HDFS : 파일을 저장 담당
   - MapReduce : 클러스터 연산 담당

3) Hive (3.1.2) :
   - HDFS내의 파일 R/W를 SQL 방식으로 접근

4) Observer Namenode :
   - Load balancing으로 부하를 분산
   - Active Namenode 기능 중 Read 기능을 담당

2) 벤치마킹 툴

- **Teragen :** 사용자가 원하는 용량만큼의 데이터를 HDFS에 생성. MapReduce 연산으로 작동
- **Terasort :** Teragen으로 생성한 데이터를 sort(정렬). MapReduce 연산으로 작동
- **ManageEngine Application :** 실제 클러스터의 부하량을 측정하기 위한 시각화 툴

*** 설정이유**

Teragen : Terasort를 위해 필요. Namenode가 Client 요청을 받으므로 많은 양의 요청과 부하를 주기위한 대용량 데이터가 필요

Terasort : Data를 Read하는 과정에 초점을 맞추어서 Observer mode 적용유무에 따른 부하량 차이 확인

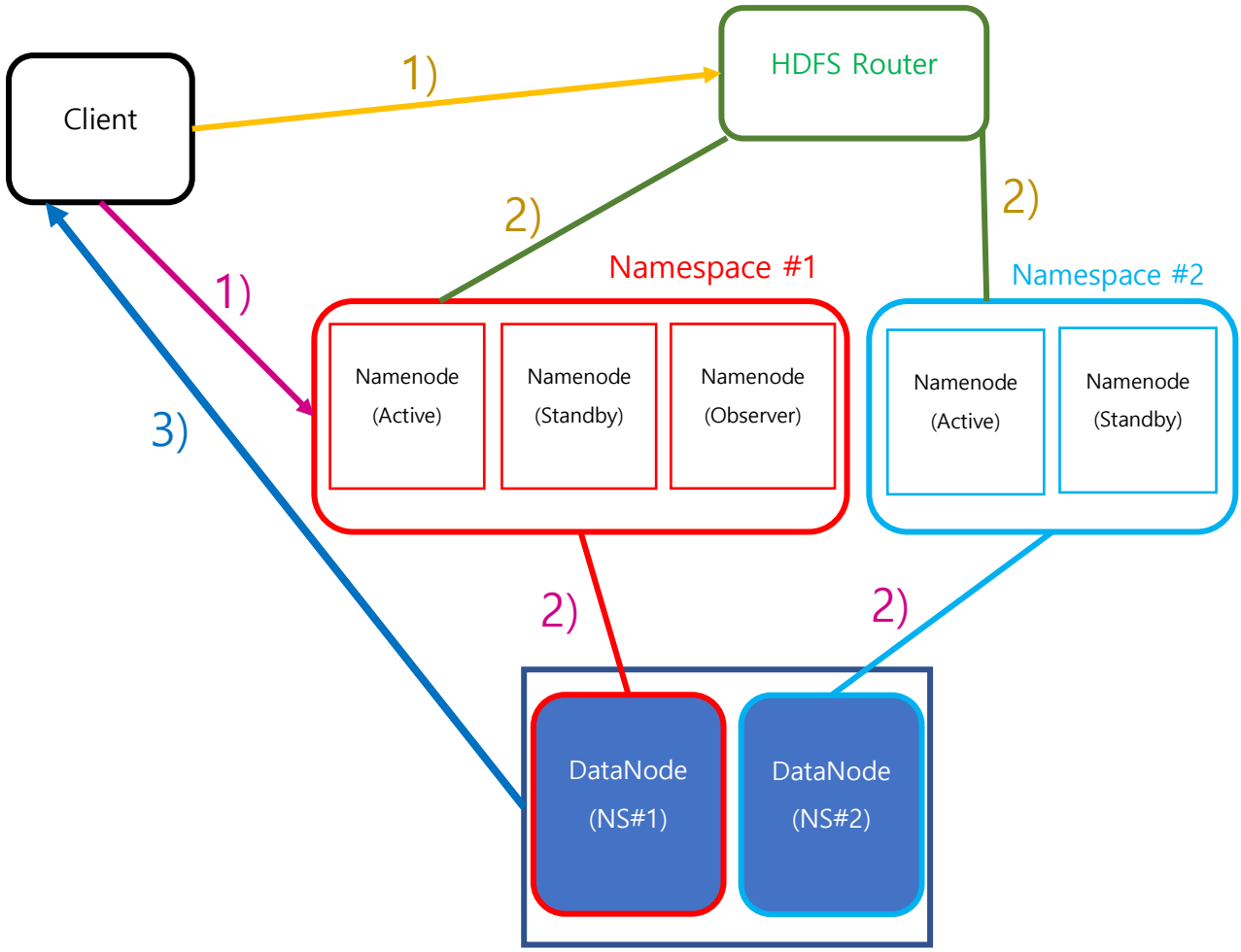
[4]**ManageEngine :** 클러스터의 현 상태를 그래프를 나타냄으로써 즉각적인 변화량 확인

3) 테스트 결과

- Observer Mode 적용 할 경우, 부하량 13% 감소, Response Time 2.78배 빠름
- Observer Mode + Mapper와 Reducer 미 지정 시, 부하량 차이 없음, Response Time 5% 빠름

2. (2) 테스트 실행 – 확장성 검증

1) 클러스터 구조



*** Multi Namespace 접근 방법**

1) Router에 접근
2) 원하는 Namenode(Namespace)에 접근 후
   메타데이터를 바탕으로 Data 위치 파악
3) 실제 Datanode에 접근하여 R/W 시행

*** 단일 Namespace 접근 방법**

1) 직접 Namenode(Namespace)에 접근
2) 메타데이터를 바탕으로 Data 위치 파악
3) 실제 Datanode에 접근하여 R/W 시행

2) 테스트 시나리오

시나리오

1) NS#1

2) NS#2

3) NS#1 + HDFS Router

4) NS#2 + HDFS Router

5) NS#1 + NS#2 + HDFS Router vs NS#1

시나리오 구성 이유

1) ,2) : 단일 Namespace 간 table join 시간 비교

3), 4) : Router를 통한 Namespace 접근을 통해 1), 2)와의 시간 비교

5) : 각 namespace에서 table을 불러 join한 시간을 단일 namespace 일 때의 경과시간과 비교

사용한 데이터 (Hive table)

[5]

movies.csv (1GB) : 영화에 대한 정보들을 모은 데이터

ratings.csv (1GB) : 영화의 평점을 모아 놓은 데이터.

Namespace 경로에 맞는 Hive Database을 생성

3) 테스트 결과

- NS#1 + NS#2 + HDFS Router이 단일 NS#1 환경일 때보다 5% 느림

- NS#1 + HDFS Router이 단일 NS#1 환경일 때 보다 2% 느림

=> Router를 통한 Namespace 접근과 그 이외의 다른 서비스 Daemon 존재로 인해 delay가 발생한 것으로 추정

3. 종합결과

- Observer Namenode의 부하 감소는 클러스터의 안정성을 높여줄 가능성 확인

- Router에 의한 소량의 delay만 존재할 뿐,
  Mutli Namespace 환경에 따른 성능 향상에 대해 의미 있는 결과는 도출되지 않음

4. 참고

[1]
http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/ObserverNameNode.html
(Observer Namenode 관련 Hadoop 문서)

[2]
https://qiita.com/hayanige/items/2fabbaa0e8fc78c8b364
(Router based Federation 설정 방법)

[3]
https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs-rbf/HDFSRouterFederation.html
(Router based Federation 설명 문서)

[4]
https://www.manageengine.com/products/applications_manager/
(Manageengine 공식 사이트)

[5]
https://towardsdatascience.com/getting-started-with-hive-ad8a93862f1a
(Hive tutorial : movielens)