목차

GlusterFS는 무엇인가? 3
개요 3
HDFS 3
문제점 4
Gluster File System(GlusterFS) 5

GlusterFS는 무엇인가?

출처 : http://jmlab.tistory.com/18

개요

Metadata 서버가 없는 분산 파일 시스템 Gluster File System

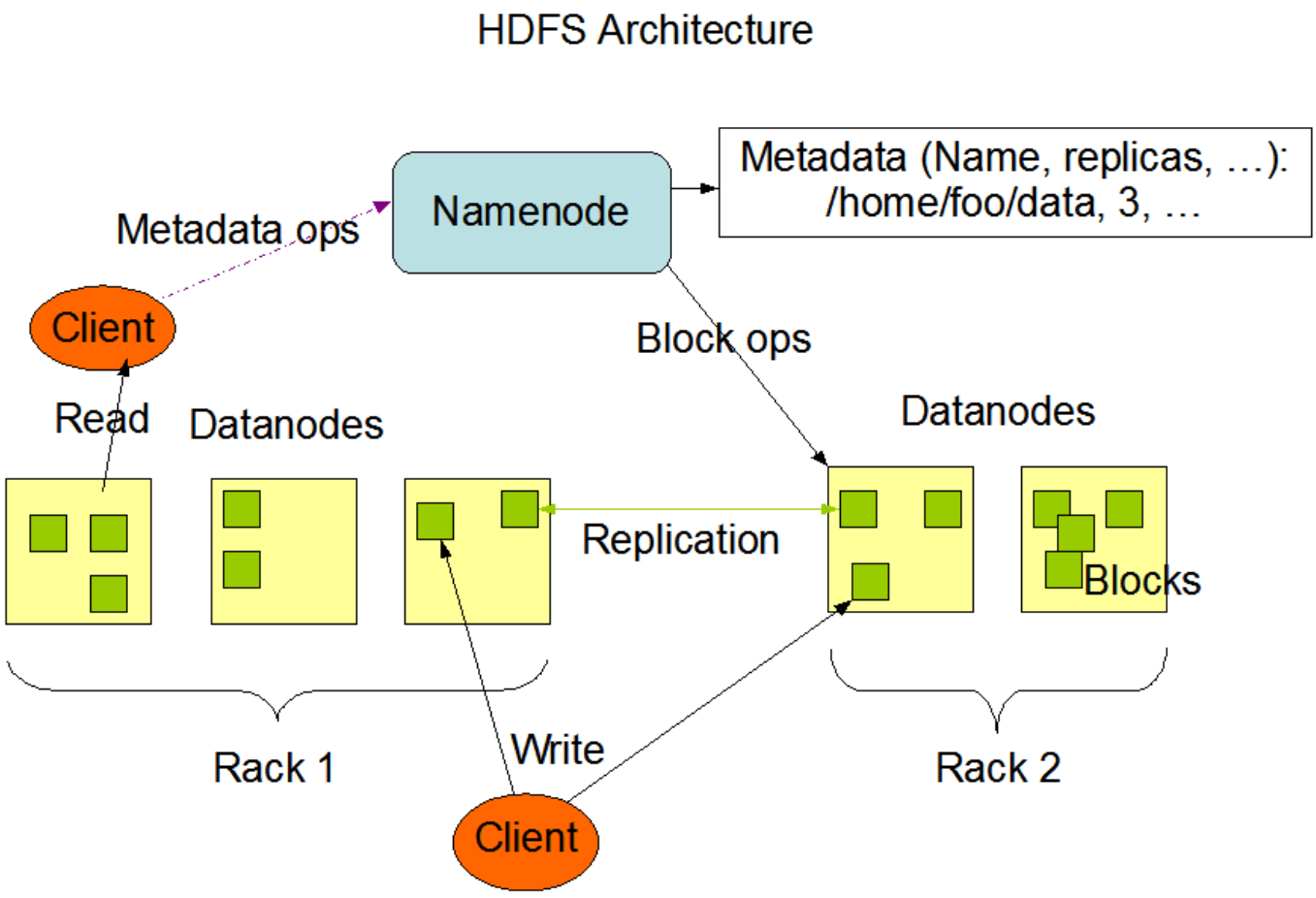
Gluster File System에서는 기존의 Distributed parallel fault-tolerant file systems 보다 빠르고, True Linear Scalability, 그리고 안정성을 가진다고 한다.

왜그런지 알기위해서 기존의 시스템에 대해 알아보자

HDFS

기존의 Distributed parallel fault-tolerant file systems (HDFS)

Distributed parallel fault-tolerant file systems 으로 잘 알려진 것이 Hadoop Distributed File System(HDFS) 이다. 물론 HDFS는 MapReduce라는 병렬 계산 알고리즘을 위해 주로 사용되는 시스템이지만 파일 시스템으로 사용하는 곳도 있다. 여기서는 파일시스템으로서 HDFS를 살펴본다.



위 그림에서 보면 Datanode는 하나의 스토리지를 포함한 서버이고, Namenode는 Metadata 서버이다. 그림에는 잘 표현 되어 있지 않지만 클라이언트는 Namenode라는 Metadata 서버에 읽으려는 파일의 분산된 위치를 받아서 직접 분산되어 저장된 Datanode 서버에서 파일 조각들과 직접 접근한다.

이와 같이 Metadata Server를통해 Datanode 서버들에 흩어져 있는 파일 정보를 한 곳에서 확인할 수 있고 파일을 여러 클라이언트에서 접근하게되면 Locking 문제들도 해결한다.

이외에도 Datanode 서버 풀 관리, 볼륨 관리, 로그 관리등등 많은 일을 한다. Metadata Server는 이와같이 중앙 집중적 관리 하기위해 필요한 것이다.

쓰기의 경우에도 Namenode 서버의 어디에 저장할지를 요청하여 알려주는 곳에 파일을 block 단위(HDFS에서는 64MB)로 조각 내어 Stripe 방법으로 저장한다.

이렇게 Namenode 서버에 걸처처 분산하여 병렬로 저장하고 읽는다.

또한 block들의 여러 복제본을 만들어(HDFS는 3개로) Datanode가 다운되더라도 다른 Datanode에 있는 복사본 Block이 원본의 역활을 대신한다.

따라서 이와 같은 아키텍처를 사용하여 Datanode 서버만 증가하면 용량과 총 처리능력이 선형적으로 증가할 수 있다.

대부분의 Distributed parallel fault-tolerant file systems 은 이런식으로 동작한다.

문제점

무엇이 문제인가?

이와같은 시스템에서 문제점들을 말할때 항상 거론되는 것이 Metadata 서버가 죽으면 어쩔 것이냐? 하는 문제가 있다.

이 문제는 대부분 Metadata 서버의 이중화로 해결하고 있다. 물론 이중화 한 것이 다 죽으면..

이것도 문제지만 만약 Cloud에 적용하는 파일 시스템에서 진짜 Linear Scalability 를 보장하느냐? 하는 문제가 있다.

상식적으로 생각해 봐도 읽고 쓰고 삭제하고 복사 할 때 마다 File등에대한 Metadata에 변경이 일어난다.

어느정도 규모에서는 이것이 큰문제가 되지 않겠지만

Cloud 환경에서는 동적으로 증가하는 수십 수백 PB의 크기의 스토리지 볼륨이 필요하고

파일들이 수십억 수백억개가 넘어(계속 증가하는)간다면 Metadata 서버가 어떻게 견딜 것인가?

물론 Metadata 서버를 확장하면 될 것인데 결국 Metadata 서버의 처리능력은

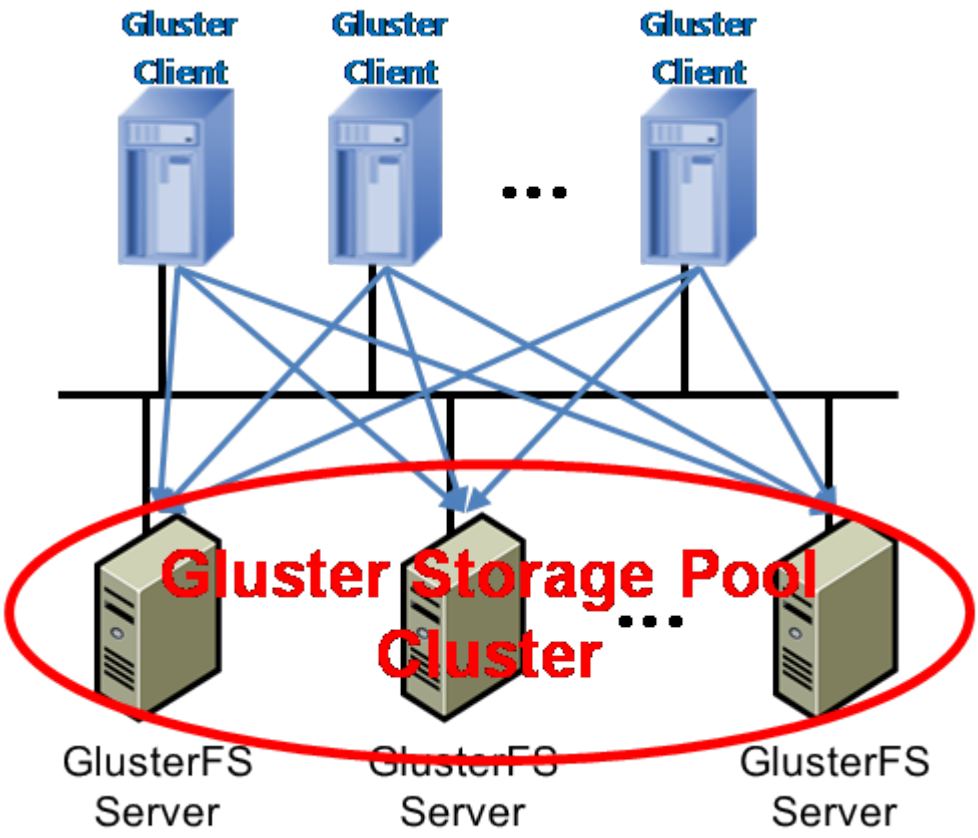
Metadata 서버들 간의 수많은 파일들에 대한 Metadata의 동기화와 수많은 파일에대한 메타데이터 요청을 처리해야하므로

결국 Metadata 서버에 의한 병목 현상이 발생하게 될 것이다. (최소한 Metadata 서버 군을 유지 하기위한 비용 및 노력이 든다)

Gluster File System(GlusterFS)

GlusterFS의 특징을 한마디로 하자면

아래 그림과 같이 Metadata Server가 없는 Distributed parallel fault-tolerant file systems 이다.



어떻게?

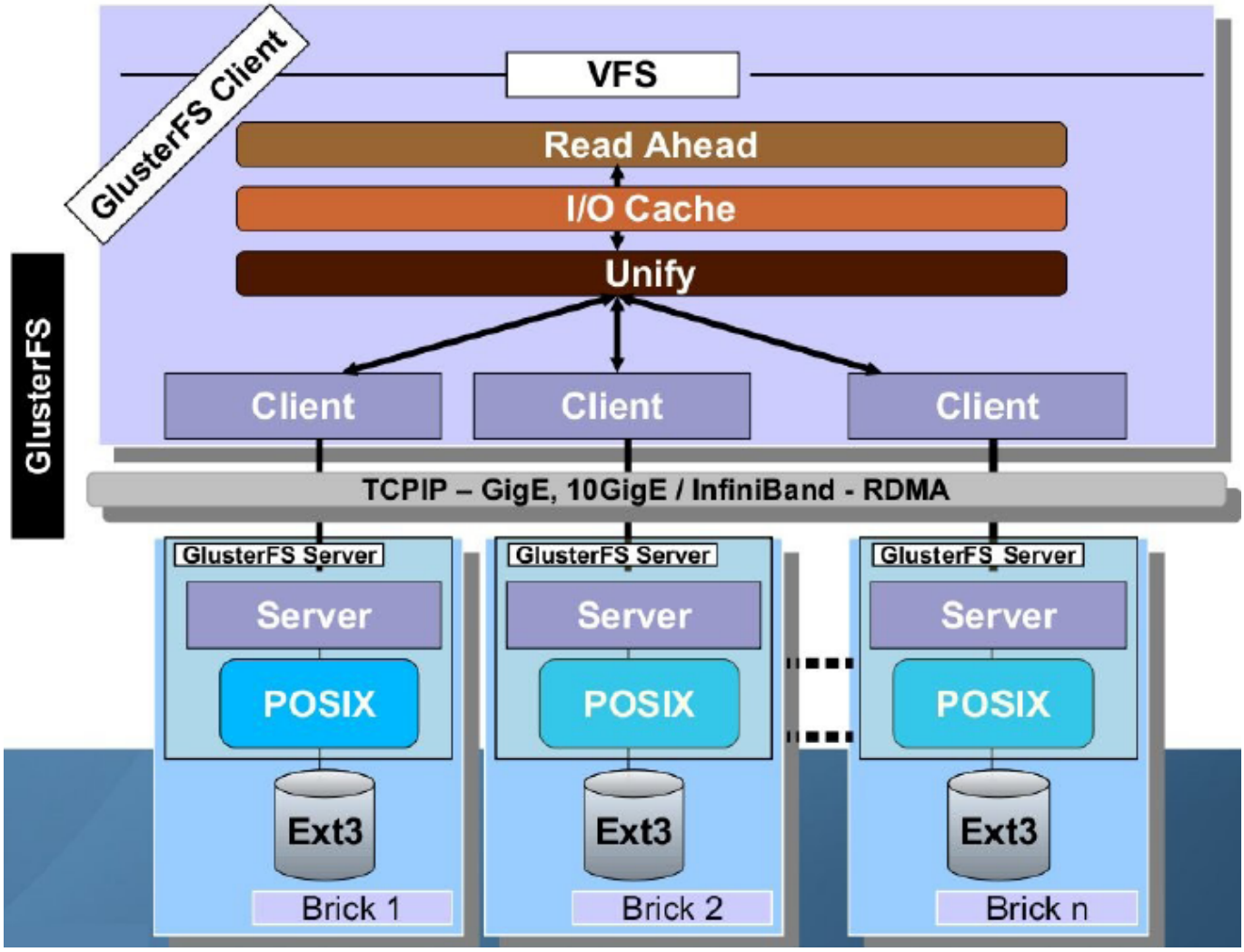
GlusterFS에는 Metadata 서버가 없다. 모든 파일들에 대한 Metadata 정보가 필요없기 때문이다.

단지 모든 서버들이 Gluster Storage Pool 에 어떤 서버들이 있고 이들을 사용한 어떤 볼륨들이 있는지만 알고 있을 뿐이다.

따라서 GlusterFS Cluster내의 어떤 서버에서도 현재 Mount 할 수 있는 볼륨의 정보를 얻을 수 있다.

그렇다면 파일 정보들을 어떻게 얻고 어떻게 사용하나?

아래 그림을 보자



GlusterFS 서버들은 자체적으로 로컬 파일 시스템을 가지고 있다.(그림에서는 Ext3)

GlusterFS에서 파일을 저장할 때 GlusterFS 서버의 로컬디스크에 지정된 폴더(Gluster에서는 이를 Brick 이라고 한다)들에

클라이언트가 보낸 파일을 변경없이 그대로 분산(Distribute 방식)하여 저장한다.

예를 들어 파일이 10개를 3대의 GlusterFS 서버로 이루어진 볼륨의 Brick에 저장한다고 하면

3곳의 Brick에 Elastic Hash 알고리즘을 사용하여 분산 저장한다.

그리고 저장된 파일을 읽는 것은 그림과 같이 각 서버들의 로컬파일 시스템의 Brick으로 지정된 폴더에서 파일들을 읽어와

클라이언트에서 마운트시 지정한 자기 폴더를 통해 볼륨의 파일들을 보게 된다.

즉 모든 파일들의 정보는 Glusterfs 클라이언트가 Volume을 Mount하면 그때 볼륨에 속한 Glusterfs 서버의 Brick으로 지정된 폴더에서

읽어와 한 폴더내에 일종의 가상으로 파일들을 보여준다.

따라서 클라이언트도 Metadata 서버에 파일 정보를 물어볼 필요없이 자신이 마운트한 GlusterFS 볼륨의 정보만 알면 된다.

이런 정보는 볼륨에 포함된 GlusterFS 서버에 GlusterFS 클라이언트가 볼륨 마운트를 요청할때 받아오고 계속 그 그서버를 통해 정보를 유지한다.

무엇이 좋은가?

Gluster File System은 Distributed parallel fault-tolerant file systems 이지만 meta-data를 사용하지 않는다.

따라서 meta-data 서버(Master Server) 가 없어 SPF 문제가 발생하지 않는다.

어떻게?

GlusterFS에는 Metadata 서버가 없다.

Gluster에서 사용될 Datanode 서버들(Glusterfs 서버)의 Pool을 Trusted Stroage Pool 이라고 한다.

GlusterFS 서버들은 모두 동일한 Metadata를 가진다. 단 포함되는 메타데이터 정보는 아래의 것 정도이다.

- Trusted Storage Pool에 포함되어 Glusterfs Cluster를 구축하는 스토리지 서버들의 정보(IP, Hash ID, 상태등)
- Trusted Storage Pool에 포함된 서버들에 걸쳐 있는 볼륨 정보 (볼륨 이름, 볼륨타입, 포함됨 서버에서 공유된 폴더 등)
- 기타 Glusterfs 시스템 로그들

여기에는 개별 파일들에 대한 정보가 없다. 사실 Metadata 서버에서 유지하는 정보 중 거의 대부분은 스토리지에 저장된

개별 파일들에 대한 정보이다. 이런 파일들에 대한 정보가 없기때문에 모든 서버가 Metadata를 가지고 있어도 큰부담이 없다.

또한 GlusterFS Cluster내의 어떤 서버에서도 현재 Mount 할 수 있는 볼륨정보를 얻을 수 있다.

From:
https://atl.kr/dokuwiki/ - **AllThatLinux!**

Permanent link:
https://atl.kr/dokuwiki/doku.php/glusterfs%EB%8A%94_%EB%AC%B4%EC%97%87%EC%9D%B8%EA%B0%80

Last update: **2017/01/03 05:31**