15장구글드라이브설계

1단계 - 문제 이해 및 설계 범위 확정

- 중요한 기능 : 업로드/다운로드/파일동기화/알림
- 플랫폼 : 모바일/웹
- 파일암호화
- 10GB 크기 제한
- DAU기준 1000만

비기능적 요구사항 : 안정석, 빠른 동기화 속도, 네트워크 대역폭, 규모확장성, 높은 가용성

개략적 추정치 free storage = 10GB

- 필요 저장공간 총량 : 10Millions * 10GB Free Storage = 500PB
- API QPS : 10Million * 2 Upload / 24 hour / 3600 seconds = 240
- 최대 : 480

개략적 설계안 제시 및 동의 구하기

- 파일 입출력 웹서버
- 사용자 데이터를 보관하는 메타 데이터 베이스
- 파일을 저장할 저장소 시스템

API

1. 파일 업로드
    a. 종류
        i. 작은 파일용
        ii. 이어 올리기(resumable upload)
    b. 인자
        i. upload type

        ii. data
2. 파일 다운로드
    a. 인자
        i. path
3. 파일 갱신 히스토리 API
    a. 인자
        i. path
        ii. limit

한 대 서버의 제약 극복

- 여러 서버에 sharding하여 데이터를 분산처리할 것
- 클라우드 오브젝트 스토리지 사용하기
    - 여러 리전에 걸쳐 다중화하여 안정성도 확보 가능함

결론

- 로드밸런서 + 웹서버 : 손쉽게 확장하기 위해서
- 메타데이터 데이터베이스 : SPOF를 회피하고 다중화 및 샤딩 정책을 적용
- 파일 저장소 : S3를 2개 이상의 리전에 다중화

동기화 충돌

- 두 사용자가 동시에 갱신하는 경우 하나의 파일을 적어도 충돌이 발생할텐데 어떻게 해소할까?
- 별도의 파일 이름으로 저장하거나 유저에게 물어볼 수도 있겠다.

개략적인 설계안

- 블록 저장소 서버 : 하나의 파일을 여러 블록으로 나눠 파일을 저장하도록하고 고유의 해시값으 할당시키는 방식 + 블록은 최대 4MB
- 클라우드 저장소 : S3

- 아카이빙저장소 : 비활성 데이터 저장
- 알림 서비스 : 파일이 추가되거나 편집/삭제 되었을 때 알림을 보낸다.
- 오프라인 사용자 백업 큐 : 클라이언트가 접속중이 아닐 때 파일 최신상태를 확인할 수 없다면 이 큐에 두어 나중에 클라이언트가 접속할 때 동기화할 수 있게 한다.

상세 설계

블록 저장소

- 파일크기가 큰 파일들을 수정되었을 때 동기화하는 방법
  - 델타 동기화 : 수정이 일어난 블록만 동기화한다.
  - 압축 : 블록 단위로 압축해 두면 데이터 크기를 많이 줄일 수 있다.
  - 위 절차는 블록 서버에서 진행되며 최종적으로는 클라우드 저장소에 압축된 변경 블록만 저장된다.

높은 일관성 요구사항

- 메타데이터 캐시는 최종일관성 모델이라 높은 일관성을 지키는 것으로 사용하기에는 부적절하다
- 따라서 일단 ACID 데이터베이스를 사용한다.

데이터베이스 테이블 : RDBMS의 경우 아래의 형태로 구성된다.

- user < device
- user < namespace < file < file_version < block
- block <→ device

업로드 과정

1. 업로드할 파일의 메타데이터를 먼저 올리고, 파일 상태를 대기중으로 바꾼뒤, 알림서비스를 통해 각 디바이스에 전달한다
2. 업로드할 파일을 업로드하고 이를 블록 단위로 쪼갠 후 압축 및 암호화하여 저장하고, 메타데이터 상태를 변경하고, 알림서비스를 통해 통지한다.

다운로드 과정

1. 변경 내역을 요청하고, 각각의 블록을 다운로드하여 파일을 교체한다.

알림서비스

이벤트를 클라이언트에 어떻게 전달할지

1. 롱 폴링 : 드롭박스, 주기적으로 이벤트를 요청해 받는 것
2. 웹 소켓 : 지속적인 통신 채널을 제공하여 양방향 통신이 가능하도록 하는 것

저장소 공간 절약

1. 중복 제거 : 중보된 파일 블록을 계정 차원에서 제거
2. 지능적 백업 전략 : 한도 설정(파일 버전 개수 상한), 중요 버전만 보관(불필요한 사본을 제거하기 위해서 중요한 것만 보관)
3. 자주 쓰이지 않는 파일을 아카이빙 저장소로 옮김

장애처리

- 로드밸러서 : HA 구성을 통해 이중화한다.
- 블록 저장소 : 작업 중인 서버가 장애가 발생하면 다른 서버가 이를 받아 처리할 수 있도록 변경
- 클라우드 저장소 : 지역 다중화
- 기타 컴포넌트 : 대부분 주-부 모델의 다중화 방식 사용
- 알림 서비스 : 연결이 끊긴 상황에서는 롱폴링으로 전환한다. 이를 다시 상태 연결로 전환할 때까지 꽤많은 시간이 걸린다.
- 오프라인 백업 큐 도 다중화하자

마무리

- 클라이언트 보안이나 클라이언트 압축도 고려해볼법하다
- 접속 상태를 관리하는 것을 별도 서버로 구성하자.