

Week 13

# 오픈소스 소프트웨어



한성대학교 컴퓨터공학부  
한 기 준 교 수

# AWS 소개

Week 13

## 학습목표

- ▼ EBS, SSH, Elastic IP 이해하기
- ▼ Elastic Load Balancing 이해하기
- ▼ 오토 스케일링 이해하기
- ▼ S3 이해하기





# 1 EBS, SSH, Elastic IP



## 아마존 EBS (Amazon Elastic Block Store)

- ✓ 아마존에서 제공하는 영구적인 블록 스토리지 볼륨
- ✓ EC2와 인스턴스를 조합하여 사용함
- ✓ HDD와 SSD를 선택할 수 있음
  - HDD는 대용량을 지원하며 가격이 저렴하나 성능이 낮음
  - SSD는 가격이 비싸지만 IOPS (Input Output Per Second)가 빠름

 HDD	보통	읽기 쓰기 속도	빠르다
	보통	가격	높다
	조금 높다	소비 전력	보통
	악하다	내구성	강하다
	이중화되어 있다	고장	이중화되어 있다
HDD (Hard Disk Drive)			
 SSD	보통	읽기 쓰기 속도	빠르다
	보통	가격	높다
	조금 높다	소비 전력	보통
	악하다	내구성	강하다
	이중화되어 있다	고장	이중화되어 있다
SSD (Solid State Drive)			



# 1 EBS, SSH, Elastic IP



## 아마존 EBS (Amazon Elastic Block Store)

- ✓ HDD, SSD 두 형태 모두 IOPS 성능을 보장하는 유형과 스루풋에 최적화된 유형 등의 볼륨 유형이 있어 성능과 요금을 비교하여 선택할 수 있음
- ✓ 요금은 '단가 X 저장 시간' 으로 산출함
- ✓ 사용 여부와 관계없이 저장하고 있는 용량만큼 비용이 부과됨
- ✓ 서버(인스턴스)를 정지해도 요금이 부과되며, 단가는 디스크의 종류에 따라 다름
- ✓ IOPS (Input Output Per Second)가 빠름

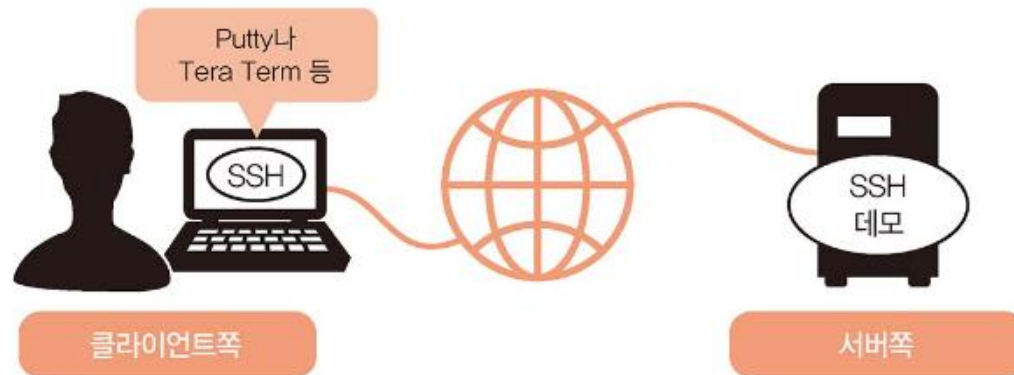
탄력적 볼륨	볼륨 크기를 간단히 조정할 수 있는 기능이다.
스냅샷	어떤 시점의 데이터를 통째로 저장하는 기능이다.
데이터 라이프 사이클 매니저	일정에 따라서 스냅샷을 생성, 삭제하는 기능이다.
최적화 인스턴스	특정 인스턴스 유형을 최적화 인스턴스로써 읽기 쓰기를 고속화하는 기능이다.
암호화	데이터 볼륨, 부팅 볼륨 및 스냅샷을 암호화하는 기능이다. KMS(AWS Key Management Service, 키를 생성/관리할 수 있는 기능)를 사용할 수 있다.

# 1 EBS, SSH, Elastic IP



## SSH

- ✓ EC2는 서버에 설치한 소프트웨어의 SSH로 관리함
- ✓ 키 페어 파일을 분실하면 서버 자체를 다시 만들어야 하기 때문에 신중한 관리가 필요함
- ✓ 서버는 SSH 사용을 위한 OS가 이미 동작하고 있으며, 클라이언트는 Putty, WinSCP등의 접속 프로그램을 활용해야 함

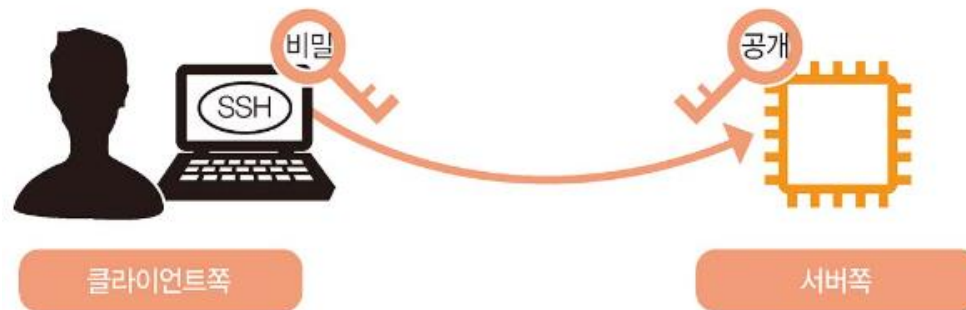


# 1 EBS, SSH, Elastic IP



## SSH

- ✓ 키 페어: 로그인할때 인증 수단으로 사용되며 공개키와 비밀키 한 쌍으로 구성되어 있음
- ✓ 공개된 키를 공개키, 자신만이 알고 있는 키를 개인키라고 하며 AWS는 이 두 키를 파일 하나로 취급함
- ✓ 인스턴스에 접속할때 인스턴스쪽에서 키 페어에 포함되어 있는 공개키를 지정하고, 클라이언트는 내려받은 키 페어 파일을 비밀키로 설정하여 사용함
- ✓ 같은 리전의 서비스라면 공통의 키 페어를 사용할 수 있음

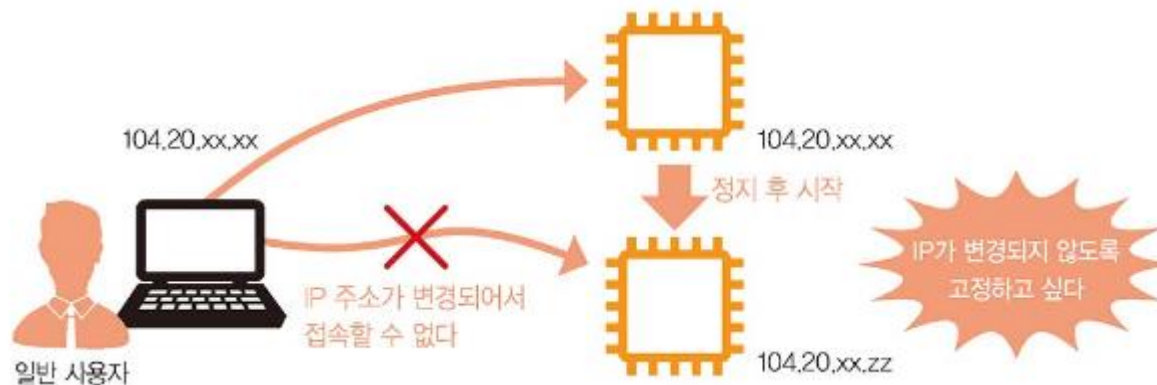


# 1 EBS, SSH, Elastic IP



## ■ Elastic IP

- ✓ AWS에는 고정 IP주소로 사용하는 Elastic IP를 제공하고 있음
- ✓ Elastic IP 주소는 AWS 계정에 속해 있기 때문에 인스턴스를 삭제해도 계속 사용할 수 있음
- ✓ Elastic IP: AWS가 제공하는 정적인 공인 (public) ip 주소임
- ✓ EC2 인스턴스는 정지 후 다시 시작하면 공인 IP 주소가 바뀌기 때문에 이를 고정하는 것이 Elastic IP임



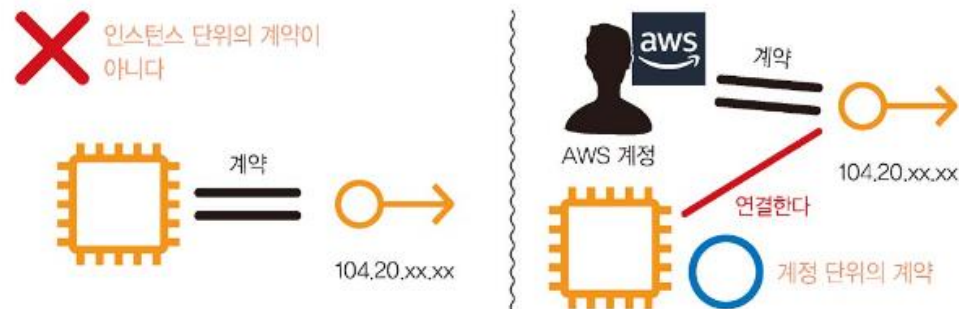


# 1 EBS, SSH, Elastic IP



## Elastic IP

- ✓ Elastic IP 주소는 AWS 계정에 연결되어 있고 인스턴스 단위가 아니기 때문에 IP주소를 부여한 인스턴스를 삭제해도 확보한 IP주소는 그대로 AWS 계정에서 소유하는 것이 가능함
- ✓ 보유한 IP주소는 다른 인스턴스나 네트워크에 부여할 수 있음
- ✓ 이미 할당되어 있는 IP주소는 Elastic IP주소로 사용할 수 없음
- ✓ Elastic IP 주소는 리전에 속하므로, 다른 리전이 보유하고 있는 Elastic IP역시 사용할 수 없음
- ✓ 기본적인 IP 한 개는 무료, 추가로 IP를 연결하면 시간에 비례하여 요금이 부과됨
- ✓ 소유한 IP가 중지된 인스턴스나 분리된 네트워크에 연결된 경우에도 시간당 요금이 부과됨

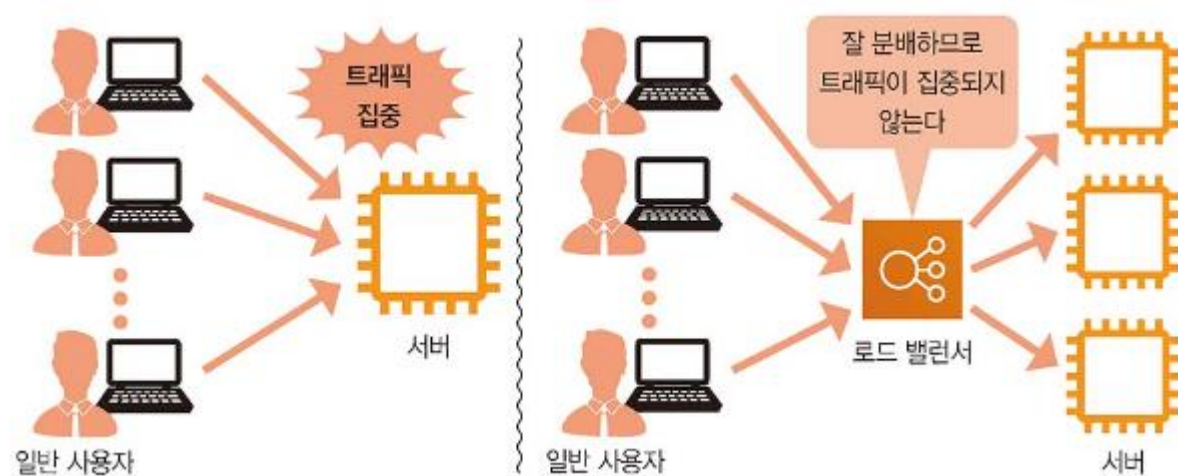


## 2 Elastic Load Balancing



### ELB

- ✓ ELB: AWS에서 제공하는 로드 밸런서
- ✓ 로드 밸런서: 서버에 집중되는 접속(트래픽)을 서버 여러 대나 네트워크에 분배하는 방식
- ✓ 부하 분산 장치: 서버 한 대에 집중되는 부하를 분산시키는 장치



## 2 Elastic Load Balancing



### Application Load Balancer (ALB)

- ✓ HTTP 및 HTTPS에 가장 적합함
- ✓ OSI 모형의 애플리케이션 계층에서 동작함
- ✓ 요청되는 명령어의 내용을 보고 판단하기 때문에 대상의 URL 디렉토리 단위로 분배하는 것이 가능함
- ✓ 인스턴스와 로드 밸런서 사이의 통신은 암호화가 가능함
- ✓ 정적 IP 주소를 설정하고, 그 IP를 가진 호스트로 전송할 수 없음

## 2 Elastic Load Balancing



### ■ Network Load Balancer (NLB)

- ✓ OSI 모형의 전송 계층에서 동작함
- ✓ 패킷이라고 불리는 단편 데이터밖에 볼 수 없어 ALB만큼 상세하게 분배할 수 없음
- ✓ 분배 대상의 정적 IP주소를 설정할 수 있고, 서버에 접속한 클라이언트의 IP주소를 그대로 서버에 전송하도록 설정할 수 있음
- ✓ 지원 프로토콜: TCP, TLS

## 2 Elastic Load Balancing



### ■ Classic Load Balancer (CLB)

- ✓ 오래된 유형의 로드 밸런서
- ✓ 지원하는 프로토콜이 많음
- ✓ 지원 프로토콜: TCP, SSL/TLS, HTTP, HTTPS

## 2 Elastic Load Balancing



### ELB의 요금

- ✓ ALB와 NLB의 요금은 시간당 사용 요금과 LCU(로드 밸런서 용량 단위) 요금의 합계로 계산함
- ✓ 사용 요금: 사용 단가 X 시간
- ✓ LCU 요금: LCU사용량 X LUC 단가 X 시간
- ✓ LCU의 4가지 항목
  - 새 연결 수 (1LCU = 초당 25개의 새로운 연결)
  - 활성 연결 수 (1LCU = 분당 3,000개의 활성 연결)
  - 처리된 바이트 (1LCU = 시간당 1GB)
  - 규칙 평가 (1LCU = 초당 1,000개의 규칙 평가)

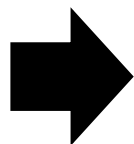


## 2 Elastic Load Balancing



### ELB의 요금 산출 예

항목	예	LCU로 환산한 결과
새로운 연결 수	평균 1개/1초	0.04 LCU
활성 연결 수	120개의 활성 연결 수 /1분	0.04 LCU
처리 바이트	평균 1.08GB 데이터의 전송/1시간	1.08 LCU
규칙 평가	최대 250개/1초	0.25 LCU



ALB · NLB 요금 = ① 사용 요금 + ② LCU 요금

사용 단가 X 시간

LCU 사용료 X LCU 단가 X 시간

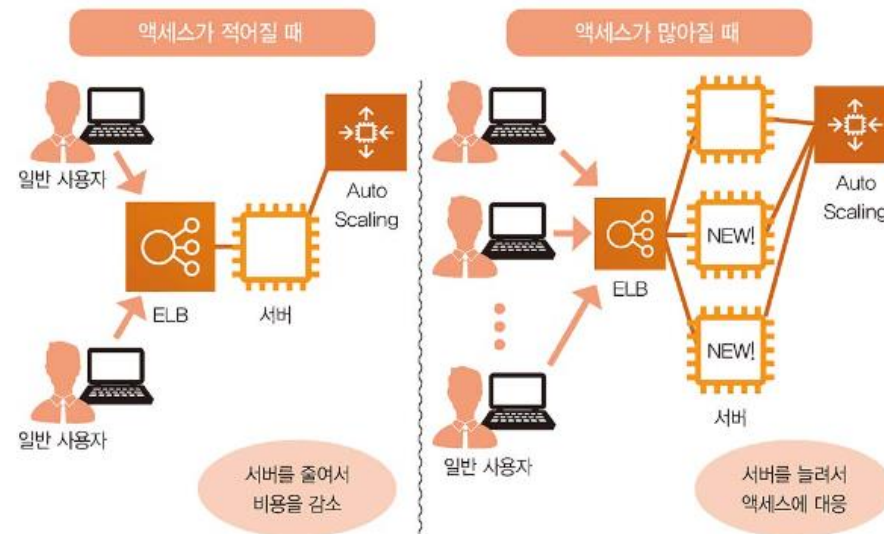
- 새로운 연결 수 0.04 LCU
- 활성 연결 수 0.04 LCU
- 처리 바이트 1.08 LCU
- 규칙 평가 0.25 LCU

이것이 최대치이므로 이 수치를 사용



## 오토 스케일링

- ✓ 오토 스케일링: 서버의 액세스 상태에 따라 서버 대수를 늘리거나 줄이는 기능
- ✓ EC2 외의 서비스를 지원하는 오토 스케일링도 있음
- ✓ AWS는 EC2 오토 스케일링을 단독으로 사용할 뿐만 아니라 서버의 부하 정보 (CPU 부하, 네트워크 통신량 등) 데이터를 참조하여 스케일링에 참고할 수 있음



# 3 오토 스케일링



## ■ 감시와 인스턴스 수의 결정

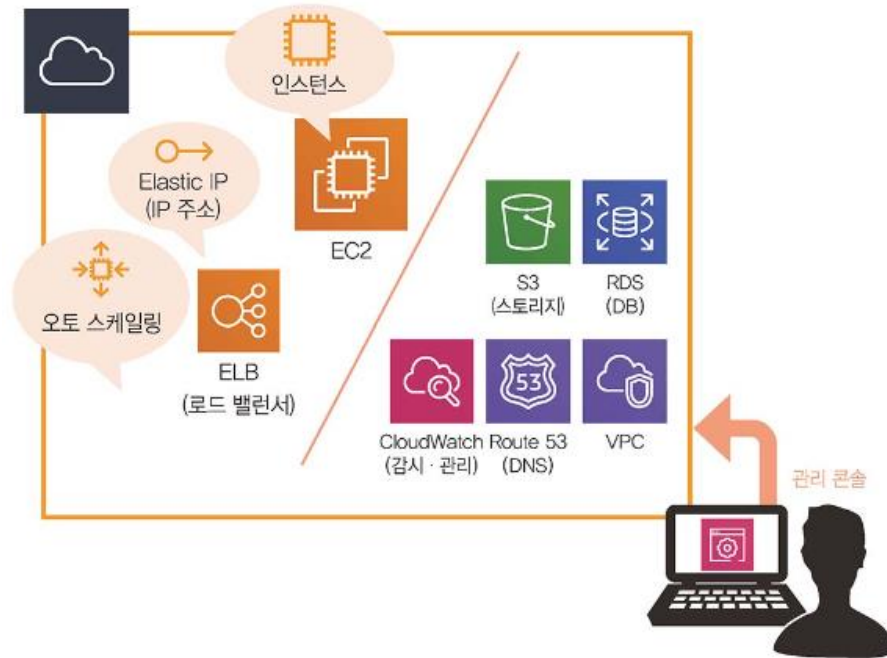
- ✓ 오토 스케일링을 시작하려면 Auto scaling 그룹을 생성하고 그룹에 인스턴스의 최소 대수와 최대 대수를 설정해야 함
- ✓ 서버 시작에 필요한 AMI, 키 페어, 보안 그룹을 설정해야 함
- ✓ 인스턴스의 증감 방법
  - EC2 인스턴스가 정지한 경우에 분리하고 새로운 EC2 인스턴스를 생성하는 방법
  - 일정에 맞춰 스케일링하는 방법
  - CPU와 네트워크의 부하를 참고하여 특정 임계 값을 넘을 때 인스턴스 수를 자동적으로 증감하는 방법
- ✓ 오토 스케일링 요금은 무료이나, CloudWatch를 사용해 서버의 부하 정보를 측정하는 경우에는 모니터링과 관련된 요금이 부과됨

### 3 오토 스케일링



#### 조합이 쉬운 EC2

- ✓ Elastic UP, EBS, ELB 등과 다른 AWS 서비스와 조합하여 시스템 전체를 구축함



항목	내용
S3	인터넷용 스토리지 서비스
RDS	관계형 데이터베이스 서비스
CloudWatch	모니터링 관리 서비스
Route 53	DNS 서비스
VPC	가상 사설 클라우드

# 3 Amazon S3



## Amazon S3

- ✓ 객체 스토리지: 데이터를 객체 단위로 관리하는 형식
- ✓ 용량 제한이 없어 최소한의 용량으로 시작 가능
- ✓ S3 장점
  - 간편한 웹 서버를 만드는데 사용할 수 있음
  - 확장 및 축소가 간편함
  - 쿼리 기능으로 집계할 수 있음

# 3 Amazon S3



## Amazon S3 특징

- ✓ 확장성: 사용 목적에 맞게 다양한 스토리지 클래스가 준비되어 있고 수명주기 정책을 사용하여 자동으로 이동이 가능함
- ✓ 가용성: 99.99%의 데이터 내구성 확보를 위해 최소 4개 가용 영역에 자동으로 복제되어 보존되기 때문에 장애나 오류, 위협에 강함
- ✓ 신뢰성: 암호화 기능, 접근 관리 도구, 감사 기능을 갖추고 있음
- ✓ 다양한 관리 기능: 스토리지 클래스 분석, 수명 주기 정책과 같은 관리 기능이 있음
- ✓ 스마트 기능: S3 select라는 데이터에 쿼리를 실행하는 기능이 있으며, AWS의 분석 서비스와 호환됨

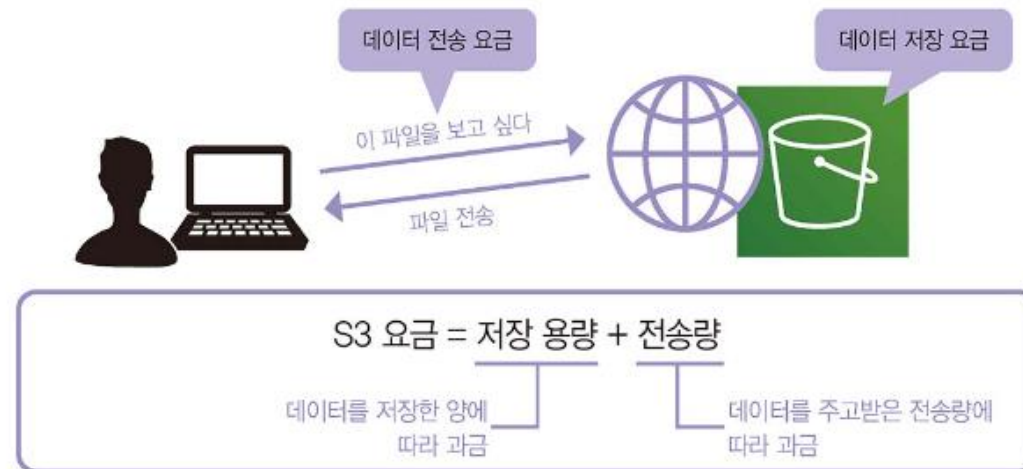


# 3 Amazon S3



## Amazon S3 요금 체계

- ✓ 보유하고 있는 용량과 전송량을 기준으로 종량 과금됨
- ✓ 저장 용량: 스토리지 클래스에 따라 일할, 30일, 90일, 180일 단위로 계산할 수 있음
- ✓ 전송량: S3에서 파일을 받거나 발생하는 요금으로, 수신 요청(GET)이나 송신 요청(PUT)에 대하여 1GB 단위로 과금됨

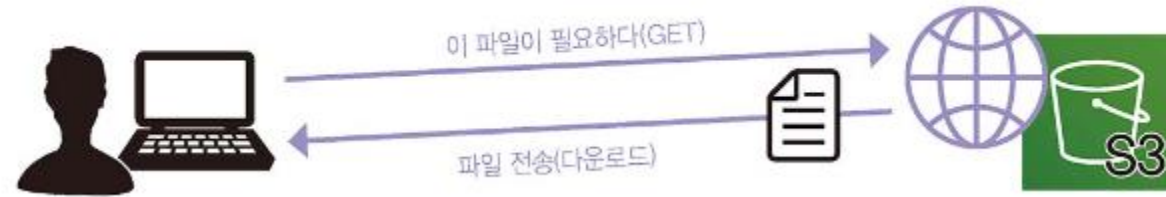


# 3 Amazon S3

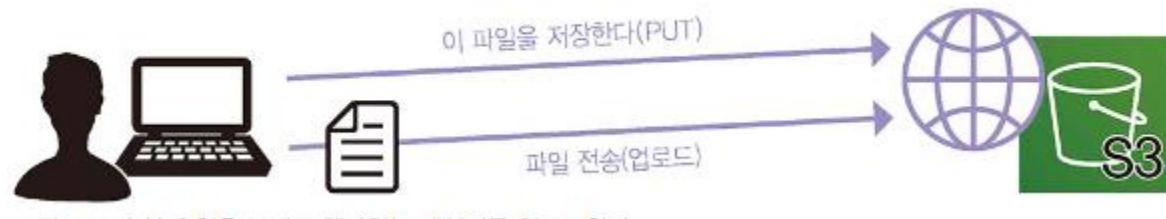


## Amazon S3 전송량 개념

- ✓ 수신 요청과 다운로드 (GET): 서버에 보내는 다운로드 명령



- ✓ 송신 요청과 업로드 (PUT): 서버에 파일을 전송하는 명령





## Amazon S3 스토리지

- ✓ 스토리지 클래스: 스토리지의 종류
- ✓ 상황에 따라 변경할 수 있음
- ✓ 스토리지 클래스의 종류
  - Standard: 데이터를 검색할 때의 요금과 최소 용량 요금이 없고 일할로 계산되어 쉽게 사용하기에 좋은 클래스
  - Intelligent-Tiering: 빈번한 액세스와 간헐적 액세스에 최적화된 두 가지 계층에 객체를 저장
  - Infrequent Access: Standard에 비해 저장 요금이 낮고 액세스 요금이 높게 설정되어 있어 액세스 빈도가 낮고 용량이 큰 데이터에 적합함
  - Reduced Redundancy Storage: 이중화 수준을 낮춰 낮은 가격으로 제공하는 클래스
  - S3 Glacier: 데이터 아카이브와 장기간 백업을 위한 스토리지 클래스이며 볼트라는 컨테이너에 저장되기 때문에 저장된 데이터를 읽기 위해서는 다른 S3 버킷으로 옮겨야 함

# 3 Amazon S3



## Amazon S3 용어

항목	내용
객체	S3의 엔티티 단위로 텍스트나 이미지 파일을 의미함
버킷	객체를 저장하는 컨테이너
버킷명	S3 버킷의 명칭은 다른 AWS 사용자를 포함하여 유일한 이름이어야 함 (웹 서버로 사용하는 경우는 도메인명이 버킷명이 됨)
객체 키	객체 식별자로, 모든 객체는 반드시 한 개의 키를 가짐. 버킷, 객체 키, 버전을 조합하여 객체를 고유하게 식별함
객체 메타데이터	이름과 값의 세트로 객체를 업로드할때 설정 가능. 사람이 파일을 쉽게 관리하기 위한 데이터
리전	버킷의 물리적인 보관 장소가 있는 지역
데이터 일관성	가용성 유지를 위해 데이터의 무결성을 보장함
버전 관리	여러 버전을 보관하는 것. 다른 버전은 별도의 객체를 취급하는 것이 가능함
로그	버킷 단위나 객체 단위의 로그를 기록할 수 있음
암호화	S3에 저장되는 데이터를 자동으로 암호화할 수 있음
액세스 제어	S3 버킷에 대한 권한을 설정함
웹 서비스	S3 버킷을 웹 사이트로 사용하는 기능

# 3 Amazon S3



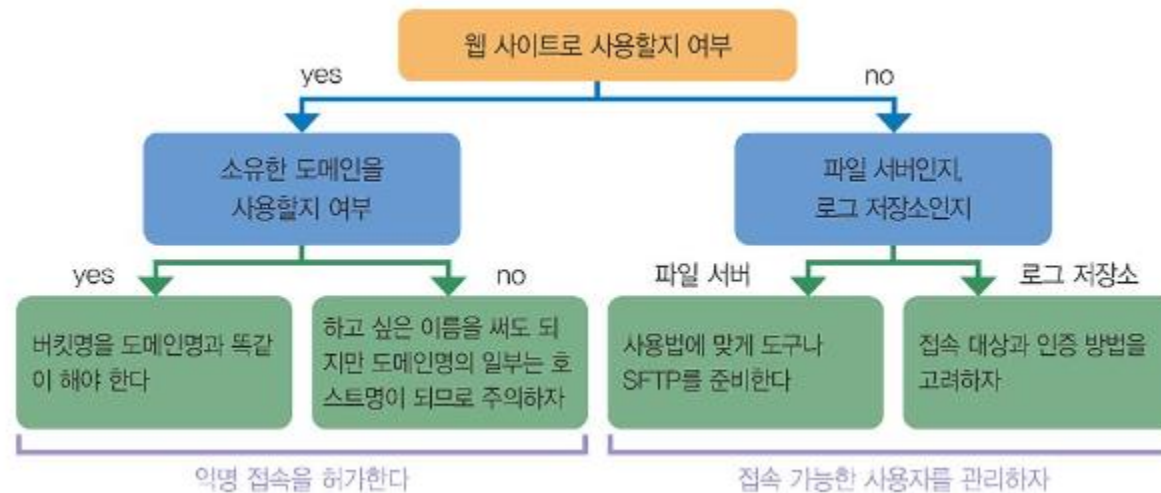
## Amazon S3 사용 절차

- ✓ AWS에 로그인: 리전을 선택하고 S3 대시보드에 접근함
- ✓ 버킷 생성: 버킷명을 설정함
- ✓ 버킷 설정: 웹 서버로 사용할 경우 static website hosting을 설정하고 접근할 수 있는 사용자를 설정함
- ✓ 파일 업로드: 관리 콘솔이나 전용 도구에서 파일을 업로드함



## Amazon S3 유의 사항

- ✓ S3 버킷을 생성한 후에는 이름과 리전을 변경할 수 없음
- ✓ 사용 용도를 명확하게 설정해야 함
- ✓ 버킷 생성 전에 아래와 같은 사항을 점검해야 함

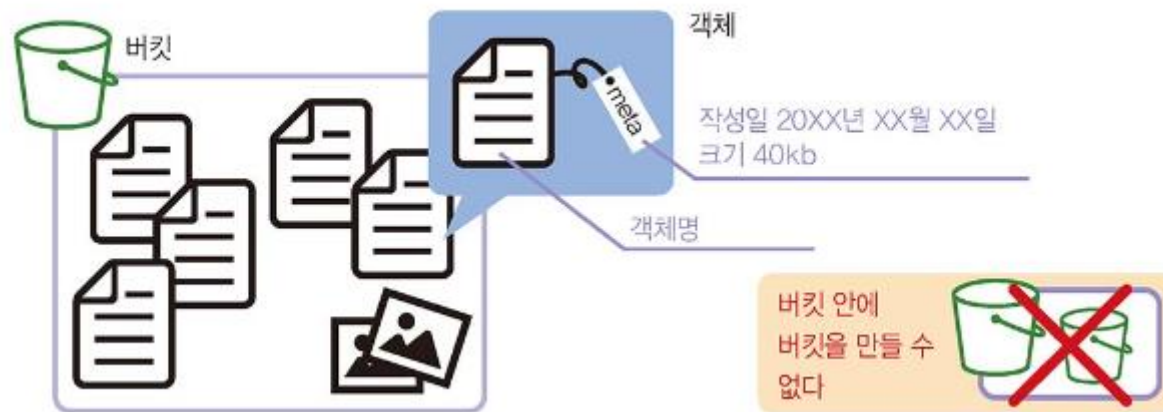






## ■ 객체와 버킷

- ✓ 버킷은 윈도의 드라이브, 객체는 파일과 같음
- ✓ 버킷은 폴더가 아니기 때문에 버킷 안에 버킷을 만드는 것은 불가능함
- ✓ 버킷 한 개에 저장할 수 있는 객체수는 제한이 없고 총 용량에도 제한이 없음
- ✓ 객체는 버킷에 계층 구조가 아닌 병렬로 배치되나 사용자 편의를 위해 폴더로 표시됨





## 버킷 생성과 명명 규칙

- ✓ 버킷명은 S3 안에서 유일해야 함
- ✓ 다른 사용자가 쓰고 있는 버킷명은 사용할 수 없음
- ✓ 버킷의 명명 규칙

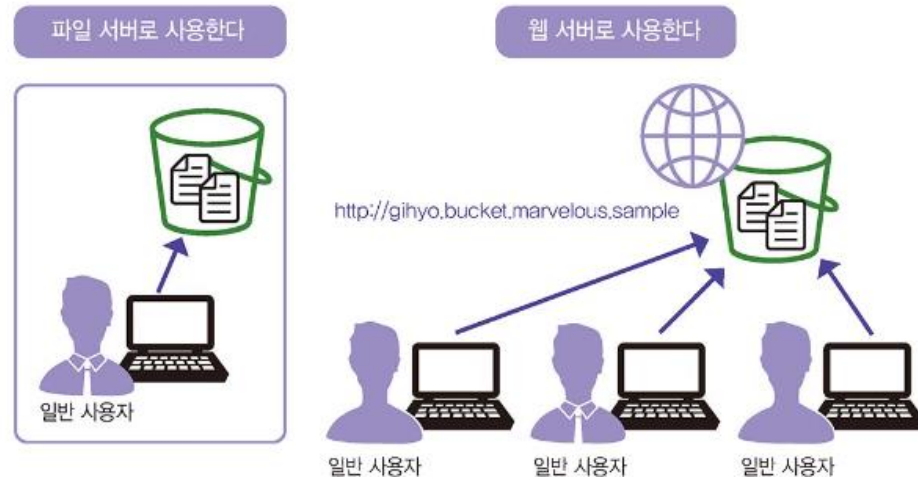


# 3 Amazon S3



## ■ S3를 통한 웹 사이트 호스팅

- ✓ 정적 웹 사이트를 호스팅할 수 있음
- ✓ 정적 웹 사이트: 서버가 스크립트를 처리하지 않는 사이트
- ✓ 정적 웹 사이트를 호스팅하려면 버킷을 그대로 웹 사이트로 오픈하면 됨



# 3 Amazon S3



## ■ 웹 호스팅에 필요한 설정

- ✓ 정적 웹 호스팅을 활성화해야 함
- ✓ 공용 액세스 차단을 해제함
- ✓ 버킷 정책을 '모든 사용자'로 설정
- ✓ 버킷명을 사용할 도메인명으로 지정
- ✓ 개인 도메인을 소지한 경우 Route53등의 DNS서비스를 활용함
- ✓ Amazon Lightsail이나 Amplify를 활용해서도 웹 호스팅이 가능함
  - Lightsail: EC2를 단순화한 서비스로 한 가지 기능을 조합한 패키지로 제공
  - Amplify: 모바일 앱이나 웹 앱을 개발하기 위한 프레임워크



## ■ S3를 통한 업로드와 다운로드

- ✓ 관리 콘솔을 사용하거나 CLI를 사용할 수 있음
- ✓ 도구 및 프로그램에서 작업하려면 API나 SDK를 사용해야 함
- ✓ API와 SDK: 서드 파티 (third party) 도구를 사용해 파일을 업로드할 수 있음
- ✓ 멀티 파트 업로드: 객체를 여러 개로 나누어 세트 하나로 업로드 할 수 있음
- ✓ AWS Transfer for SFTP: SFTP를 사용하여 파일을 전송할 수 있는 서비스
- ✓ AWS DataSync: 온프레미스 스토리지 시스템과 AWS 스토리지 서비스 간에 대용량 데이터 전송을 위한 서비스
- ✓ 하이브리드 클라우드 스토리지 서비스: S3 버킷을 온프레미스의 스토리지처럼 사용함



## ■ S3 버전 관리

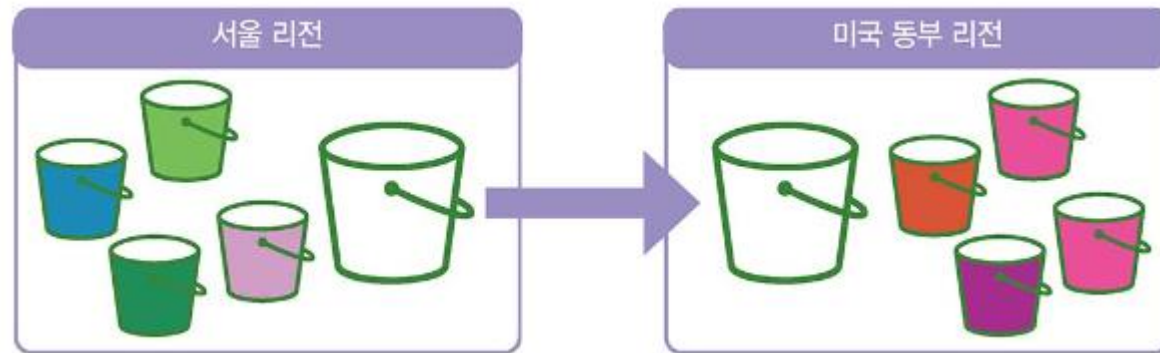
- ✓ 버전 관리: 객체를 여러 버전으로 저장하는 기능으로, 버킷 단위로 설정함
- ✓ 미사용, 활성화, 버전 일시 중지 상태 중 하나로 설정할 수 있음
- ✓ 수명 주기 정책: 객체가 정기적으로 수행할 작업을 설정함
  - Transition: 객체를 다른 스토리지 클래스로 이동함
  - Expiration: 유효 기간이 만료된 객체를 삭제함
  - NoncurrentVersionTransition: 현재 스토리지 클래스에서 객체의 유지 시간을 지정함
  - NoncurrentVersionExpiration: 과거 버전의 객체를 삭제하기 전 유지할 시간을 지정함
  - AbortIncompleteMultipartUpload: 멀티 파트 업로드 진행 상태를 유지할 최대 시간을 지정함
  - ExpiredObjectDeleteMarker: 만료된 객체 삭제 표시를 제거함





## 교차 리전 복제

- ✓ 교차 리전 복제: 다른 리전의 버킷에 객체를 비동기적으로 복사함
- ✓ 복제에 사용할 두 커밋은 버전 관리가 활성화되어야 함
- ✓ 같은 리전에 존재하는 버킷은 복제 설정을 할 수 없음
- ✓ 교차 리전 복제를 사용하면 해외에도 백업해 둘 수 있어 이중화가 향상됨
- ✓ 해외 데이터 보존의 경우 국외 반출이 안되는 데이터를 보존하지 않도록 유의해야 함





## 데이터 분석과 연계

- ✓ S3의 객체나 객체의 내용에 대한 데이터를 분석할 수 있음
- ✓ S3 select나 Amazon Athena는 CSV와 같이 구조화된 텍스트 형식의 데이터를 SQL의 SELECT 문으로 실행할 수 있음
- ✓ Amazon Redshift spectrum도 비슷한 기능을 갖고 있지만 대용량 데이터를 처리하기 때문에 데이터 웨어하우스를 제공하는 redshift cluster가 추가로 필요함

