

1주차 문제풀이

1. 어떤 회사는 Amazon S3에 회계 기록을 저장해야 합니다. 기록은 1년 동안 즉시 액세스할 수 있어야 하며, 그 후 9년 동안 보관해야 합니다. 관리자와 루트 사용자를 포함하여 회사의 어느 누구도 10년 동안 기록을 삭제할 수 없습니다. 기록은 최대한의 복원력으로 저장해야 합니다. 어떤 솔루션이 이러한 요구 사항을 충족할까요?

- A. S3 Glacier에 10년 전체 기간 동안 레코드를 저장합니다. 액세스 제어 정책을 사용하여 10년 동안 레코드 삭제를 거부합니다.
- B. S3 Intelligent-Tiering을 사용하여 레코드를 저장합니다. IAM 정책을 사용하여 레코드 삭제를 거부합니다. 10년 후 IAM 정책을 변경하여 삭제를 허용합니다.
- C. S3 Lifecycle 정책을 사용하여 1년 후 S3 Standard에서 S3 Glacier Deep Archive로 레코드를 전환합니다. 10년 동안 S3 Object Lock을 준수 모드로 사용합니다.
- D. S3 Lifecycle 정책을 사용하여 1년 후 S3 Standard에서 S3 One Zone-Infrequent Access(S3 One Zone-IA)로 레코드를 전환합니다. 10년 동안 거버넌스 모드에서 S3 Object Lock을 사용합니다

keyword

① S3 스토리지 클래스

② S3 Lifecycle Policy

③ Object Lock

I) S3의 스토리지 클래스

▼ 표 5-4 Amazon S3 수명 주기 정책에 따른 스토리지 변화

데이터에 대한 S3 수명 주기 정책을 이용한 스토리지 변화

스토리지 유형	S3 standard	S3 - IA	glacier
AWS 데이터 베이스 서비스	 S3 standard storage	 S3 infrequent access standard storage	 amazon S3 glacier
주요 특징	자주 액세스하는 데이터	액세스 빈도가 낮은 데이터	거의 액세스하지 않는 데이터
사용 사례	<ul style="list-style-type: none"> 주요 사용 데이터 빅데이터 분석용 데이터 작업용 임시 백업 데이터 	<ul style="list-style-type: none"> 파일 동기화 데이터 백업 재해 복구용 데이터 	<ul style="list-style-type: none"> 장기 보존용 데이터 시스템 백업

자주 접근

Standard

intelligent-
tiering

변경 가능성이
있는
데이터

in frequent
access

Standard
One AZ
가장 가깝지만
(AZ)

수정하고
자주 X

기동 영역
(개)
(복원 가능)

glacier
장기간 백업
(복원)

deep archive

설계할
데이터

접근 거의 X

2) S3 누명주기 관리

: 미사용 파일 삭제하여, S3 공간 절약
각 스토리지 클러스터 간 이동 가능

└ 수동설정

└ Lifecycle Rule을 통한 자동화

3) S3 가격지 장점

: S3에 저장된 가격의 수준 / 성능 비용비용

↳ Legal Hold : 유기적으로 장점

Retention Period : 투명기준통장

[가격인상 : 초기 인상의 청구 가능
가격변동 : 이후로는 불가]

둘다
수정
가능
(도입금)



2. 한 회사가 온프레미스에서 AWS 클라우드로 다중 계층 애플리케이션을 이전하여 애플리케이션의 성능을 개선하고자 합니다. 애플리케이션은 RESTful 서비스를 통해 서로 통신하는 애플리케이션 계층으로 구성되어 있습니다. 한 계층이 과부하되면 트랜잭션이 삭제됩니다. 솔루션 아키텍트는 이러한 문제를 해결하고 애플리케이션을 현대화하는 솔루션을 설계해야 합니다. 이러한 요구 사항을 충족하고 가장 운영 효율적인 솔루션은 무엇입니까

운영

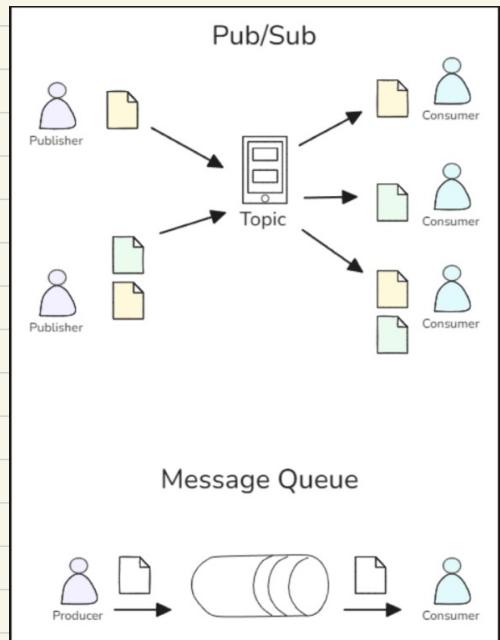
- A. Amazon API Gateway를 사용하고 AWS Lambda 함수에 대한 직접 거래를 애플리케이션 계층으로 사용합니다. Amazon Simple Queue Service(Amazon SQS)를 애플리케이션 서비스 간의 통신 계층으로 사용합니다.
- B. Amazon CloudWatch 메트릭을 사용하여 애플리케이션 성능 기록을 분석하여 성능 실패 중 서버의 최대 사용률을 확인합니다. 최대 요구 사항을 충족하도록 애플리케이션 서버의 Amazon EC2 인스턴스 크기를 늘립니다.
- C. Amazon Simple Notification Service(Amazon SNS)를 사용하여 Auto Scaling 그룹의 Amazon EC2에서 실행되는 애플리케이션 서버 간의 메시징을 처리합니다. Amazon CloudWatch를 사용하여 SNS 대기열 길이를 모니터링하고 필요에 따라 확장 및 축소합니다.
- D. Amazon Simple Queue Service(Amazon SQS)를 사용하여 Auto Scaling 그룹에서 Amazon EC2에서 실행되는 애플리케이션 서버 간의 메시징을 처리합니다. Amazon CloudWatch를 사용하여 SQS 대기열 길이를 모니터링하고 통신 오류가 감지되면 확장합니다.

운영 필요 → 효율 X

i) Messaging . 보산 어플리케이션

: JSON, 문자열 전송

- ① 발행/ 구독 패턴
- ② 메시지 큐 패턴



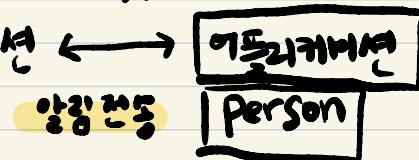
- SQS (1:1)

: 메시지 큐 서비스

메시지 누락 X 안정적 전송

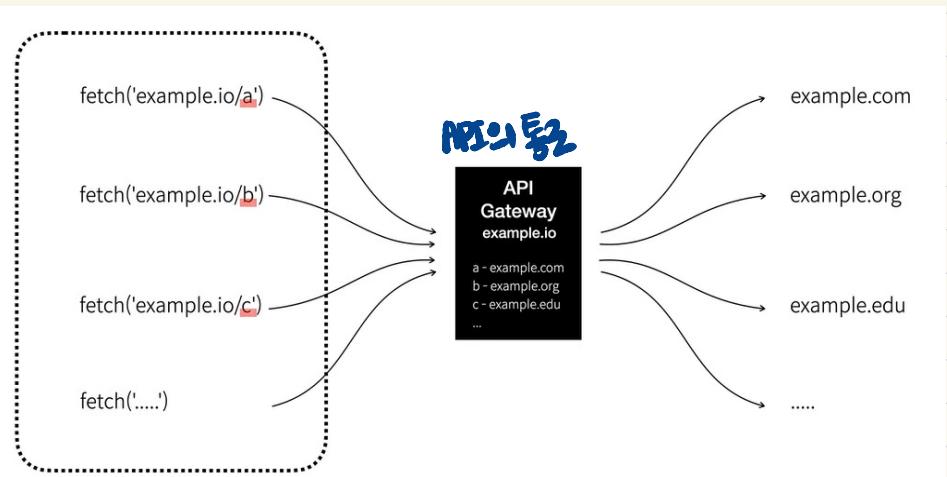
- SNS (1:N)

: 어플리케이션 ↔



ii) API Gateway

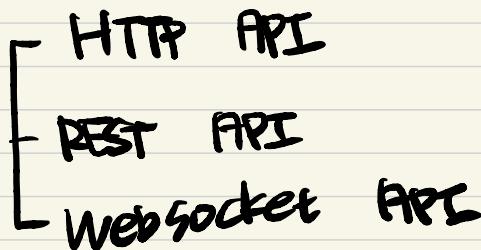
- 규모가 관계없이 API 쌍방, 유지관리, 모니터링 브로드캐스팅 서비스



· 엔드포인트와 REST API 관리 가능

- API 게이트웨이를 등록해주면, 모든 클라이언트는 각 서비스의 엔드포인트 대신 API Gateway로 요청을 전달하여 관리가 용이해 진다 (**Proxy**)

· 서버리스



iii) AWS Lambda

: 뉴베리스 컴퓨팅 상품 / 함수 바로 실행

· 자동 페스 케일링

· 배포박스

· 리소스 제한

· 상태 비저장

· Cold Start

· 동시성 제한

IV) CloudWatch

: AWS 서비스에 대한 모니터링 Metrics
dimension

EQ 엔스턴스의 CPU 점유율을 보고싶다.

namespace

자료 metric

STATISTICS
(평균 / 최대..)

3. 소셜 미디어 회사에서는 사용자가 이미지를 웹사이트에 업로드할 수 있도록 허용합니다. 이 웹사이트는 Amazon EC2 인스턴스에서 실행됩니다. 업로드 요청 중에 웹사이트는 이미지 크기를 표준 크기로 조정하고 크기가 조정된 이미지를 Amazon S3에 저장합니다. 사용자는 웹사이트에 대한 업로드 요청이 느리다는 것을 경험하고 있습니다. 이 회사는 애플리케이션 내에서 결합을 줄이고 웹사이트 성능을 개선해야 합니다. 솔루션 아키텍트는 이미지 업로드를 위한 가장 운영 효율적인 프로세스를 설계해야 합니다. 솔루션 아키텍트는 이러한 요구 사항을 충족하기 위해 어떤 조치 조합을 취해야 합니까? (두 가지를 선택하세요.)

- A. S3 Glacier에 이미지를 업로드하도록 애플리케이션을 구성합니다.
- B. 웹 서버를 구성하여 원본 이미지를 Amazon S3에 업로드합니다.
- C. 미리 서명된 URL을 사용하여 각 사용자의 브라우저에서 Amazon S3로 직접 이미지를 업로드하도록 애플리케이션을 구성합니다.
- D. 이미지가 업로드될 때 AWS Lambda 함수를 호출하도록 S3 이벤트 알림을 구성합니다. 함수를 사용하여 이미지 크기를 조정합니다.
- E. 업로드된 이미지의 크기를 조정하기 위해 일정에 따라 AWS Lambda 함수를 호출하는 Amazon EventBridge(Amazon CloudWatch Events) 규칙을 만듭니다

(Decoupling)

C: EC2 웹서버가 바이트를 중재하지 않아 결함이 많고
웹서버 부하 / 네트워크 병목 ↓ 업로드 지연 감소
서버 규모 확장 불필요

E: 스케일 배치는 오히려 관리가 어려움

Pre-Signed URL

1. 모든 파일 퍼블릭

2. IAM 자격증명 공유 (Access Key Pair)

: 지정인만 공유 가능

3. IAM 사용자 부여하기 (Role)

접근권한의 번거로움을 해결하기 위해

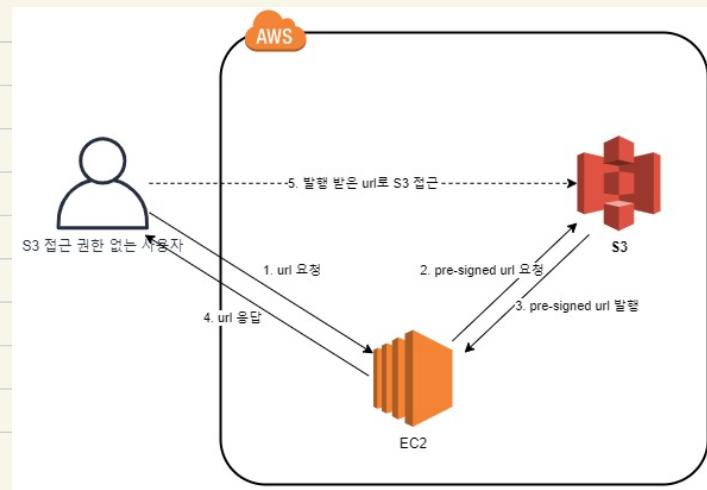
→ Pre-Signed URL

: 파일에 접근 가능한 임시 URL 생성

지정인만 공유 가능

URL 만료시간 설정 가능

Get / Post 요청 가능



4. 한 회사에서는 많은 Amazon EC2 인스턴스를 사용하여 완료하는 매우 동적인 일괄 처리 작업을 합니다. 이 작업은 본질적으로 상태 비저장이며, 부정적 인 영향 없이 언제든지 시작 및 중지할 수 있으며, 일반적으로 완료하는 데 총 60분 이상 걸립니다. 이 회사는 솔루션 아키텍트에게 작업 요구 사항을 충족하는 확장 가능하고 비용 효율적인 솔루션을 설계해 달라고 요청했습니다. 솔루션 아키텍트는 무엇을 추천해야 합니까?

- A. EC2 스팟 인스턴스를 구현합니다.
- B. EC2 예약 인스턴스를 구매합니다.
- C. EC2 온디맨드 인스턴스를 구현합니다.
- D. AWS Lambda에서 처리를 구현합니다.

최대 실행 시간 15분

EC2 : 짧은 시간에 실행환경을 구축할 수 있는 가상머신(매니지먼트 X)

수요에 따른

- 온디マン드 인스턴스 (On-demand)

: 사용한 만큼 요금 부과

- 예약 인스턴스 (RI)

: 장기간 이용시 유리 / 1년 · 3년 미리 구매 후 사용

i) Standard RI

: 동일 인스턴스 유형간 ($t2.micro, t2$) 누정 가능

재판매 가능

인스턴스 크기 변경 가능

ii) 전환기능 인스턴스 (Convertible RI)

: 할인율이 낮으나, 인스턴스 유형 / 운영체제 교체 가능
재판매 불가

- 소포트 인스턴스 (SI)

: 인스턴스 가격 제안 / 시세 변동시 인스턴스 자동 종단

AWS가 넘은 서비 발려줌

5. 어떤 회사는 확장성과 가용성에 대한 요구 사항을 충족하기 위해 컨테이너에서 중요한 애플리케이션을 실행하고자 합니다. 이 회사는 중요한 애플리케이션의 유지 관리에 집중하는 것을 선호합니다. 이 회사는 컨테이너화된 워크로드를 실행하는 기본 인프라를 프로비저닝하고 관리하는 책임을 맡고 싶어하지 않습니다. 솔루션 아키텍트는 이러한 요구 사항을 충족하기 위해 무엇을 해야 할까요

- A. Amazon EC2 인스턴스를 사용하고 인스턴스에 Docker를 설치합니다.
- B. Amazon EC2 워커 노드에서 Amazon Elastic Container Service(Amazon ECS)를 사용합니다.
- C. AWS Fargate에서 Amazon Elastic Container Service(Amazon ECS)를 사용합니다.
- D. Amazon Elastic Container Service(Amazon ECS)에 최적화된 Amazon Machine Image(AMI)에서 Amazon EC2 인스턴스를 사용합니다

1) ECS : 컨테이너 (Docker) 관리 서비스

컨테이너 관리 대상

1) EC2 IaaS

2) Fargate PaaS

c.f EKS도 컨테이너 사용인데, Kubernetes를
기반으로 한다.

Console : 클러스터 모니터링 / 제어

[구성요소] Cluster : 여러대의 컴퓨터들이 연결되어 하나로 통합되는 환경

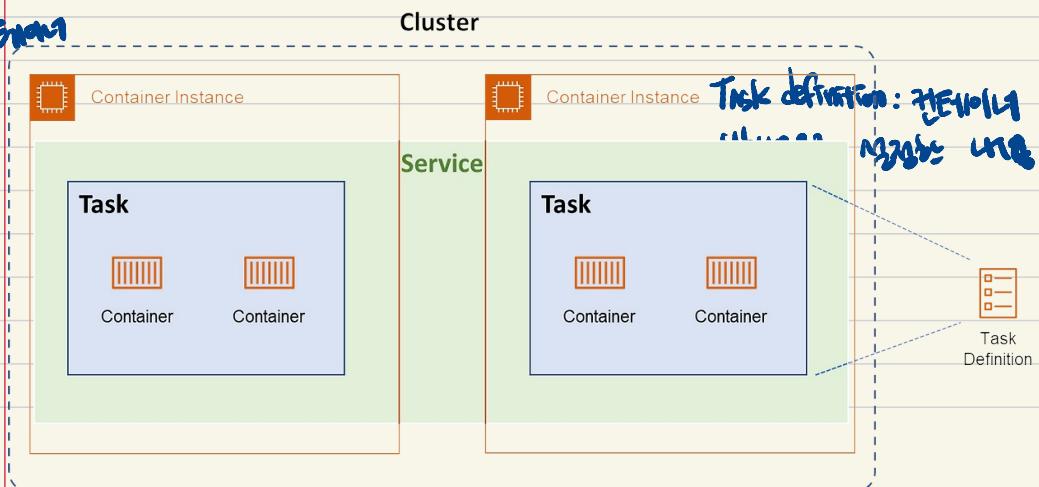
Service : Task를 실행하기 위한 서비스로 정의

Container Instance : ECS를 통하여 Task가 실행되는 EL2 인스턴스

Task : Task definition

정의된 서비스에 대한

설정값



ECS : Docker 컨테이너 관리 서비스

과거: EC2 위에 Docker화하여 서비스 구축

현재: ECS 사용하기 ↗ 이 과정이 필요 X

컨테이너 실행을 위한 인프라를 제공한다

→ ECS는 EC2 or Fargate 같은 관리형 컨테이너

플랫폼에서 실행된다

컨테이너를 위한 서비스 제공!

(AWS에서 제공하는 OS과 AWS가 제공)



등록

- 1) T-CP의
- 2) 콤보스터지의

- 3) 마운트의
- 4) Task 생성

1. ECS개요 – 비용 절감

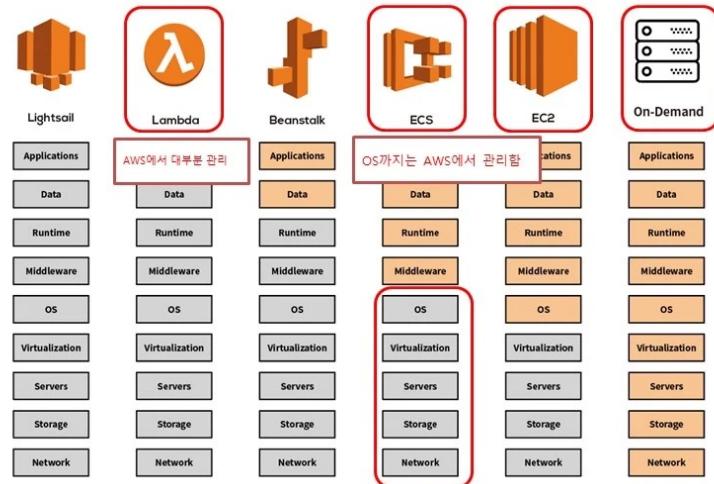
Lambda (저비용) < ECS(중간비용) < EC2(고비용)



8

2. ECS개요 – 운영부담 감소

Lightsail < Lambda < ECS < EC2 < On-Demand (운영부담)



9

AMI (Amazon Machine Image)

: EC2 인스턴스를 만들기 위한 기본 이미지

OS, CPU....

= 이미지