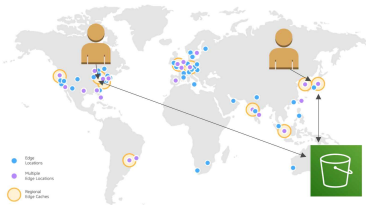
CloudFront

일시	2023.04.08. (토)	작성자	김민경

1. CloudFront

1) 개요



-Content Delivery Network(CDN:컨텐츠 전송 네트워크)

-웹사이트의 컨텐츠를 서로 다른 엣지 로케이션에 미리 캐싱하여 읽기 성능 ⇑

-현재 전세계에 있는 총 216개의 엣지 로케이션을 통해 구성됨

-컨텐츠가 전세계적으로 분산되어 있음

∴DDoS 공격에서 보호받을 수 0

*엣지 로케이션
:CloudFront의 캐싱 콘텐츠가 위치하는 곳

*DDoS 공격
:동시에 모든 서버가 공격받는 방식

2) Origin (원본제공 방식)

(1) S3 bucket

-CloudFront를 통해 파일을 분산하고 캐싱할 수 있게 함

-버킷에는 CloudFront만 접근할 수 있게 보장 (OAC(Origin Access Control: 원본 접근 제어))

⇒**OAI(Origin Access Identity) 대체**

*OAI
:CloudFront가 S3에 저장된 Private 객체에 액세스할 수 있도록 하는 특별한 식별자

*참조

Amazon S3 오리진에 대한 액세스 제한

PDF | RSS

CloudFront는 Amazon S3 오리진에 인증된 요청을 전송하는 두 가지 방법으로 *오리진 액세스 제어*(OAC)와 *오리진 액세스 ID*(OAI)를 제공합니다. OAC는 다음을 지원하므로 OAC를 사용하는 것이 좋습니다.

- 2022년 12월 이후에 출시된 옵트인 리전을 포함하여 모든 AWS 리전 리전의 모든 Amazon S3 버킷
- Amazon S3 AWS KMS를 사용한 서버 측 암호화(SSE-KMS)
- Amazon S3에 대한 동적 요청(PUT 및 DELETE)

OAI는 위 목록의 시나리오에서 작동하지 않거나 이러한 시나리오에서 추가 해결 방법이 필요합니다. 다음 주제에서는 Amazon S3 오리진에서 OAC를 사용하는 방법을 설명합니다. OAI에서 OAC로 마이그레이션하는 방법에 대한 자세한 내용은 오리진 액세스 ID(OAI)에서 오리진 액세스 제어(OAC)로 마이그레이션 섹션을 참조하세요.

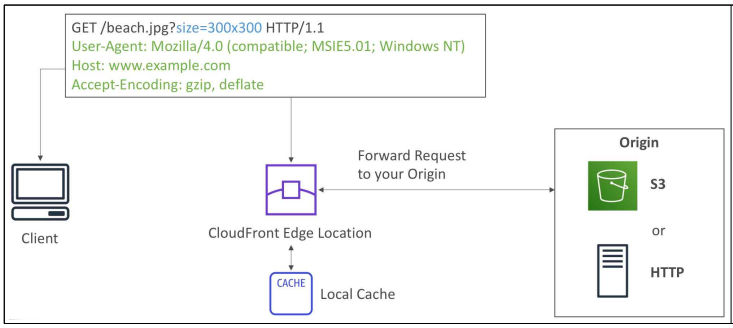
(2) Custom Origin (HTTP)

-HTTP 백엔드와 같은 사용자 정의 원본 사용 가능

-ALB, EC2 인스턴스, S3 웹사이트, 그 외의 다른 HTTP 백엔드

(단, S3 웹사이트 ⇒ 버킷 활성화해서 정적 웹사이트로 설정해야 함)

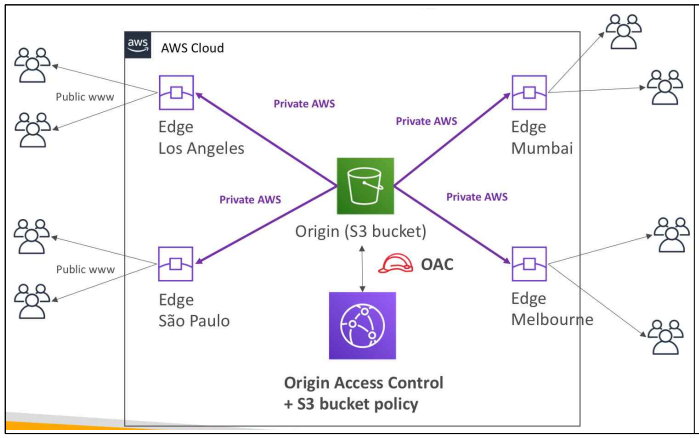
3) CloudFront의 작업방식



- client가 edge locations에 http 요청 보냄
→ edge는 캐싱 여부 확인함
→ 캐싱되어 있지 x 땐 원본에서 가져옴
→ 가저온 후엔 Local Cache에 저장

- 캐시에 저장하면 같은 요청을 받을 시 원본에서 다시 가져올 필요x

4) S3 as an Origin



- 내부 AWS 망을 통해 S3 원본 버킷을 받아옴
(이 버킷은 OAC로 보호받음,
S3 버킷 정책에 의해서만 수정 가능)

- 사용자에게 가장 가까운 엣지에서 결과 받음

- CloudFront와 Edge Location을 이용해
특정 리전에 속한 S3 버킷을 전세계의
Edge Location으로 분산 가능

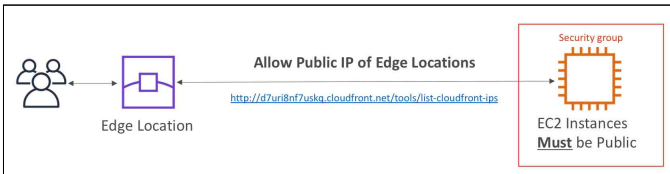
5) CloudFront vs S3 Cross Region Replication (교차 리전 복제)

(1) CloudFront : 전세계에 걸친 컨텐츠 전송 네크워크
-전세계의 Edge net 사용
-216개의 엣지 로케이션에 하루 동안 파일이 캐싱됨
-전세계를 대상으로 한 정적 컨텐츠를 사용할 시 용이

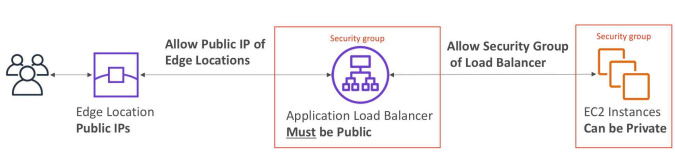
(2) S3 Cross Region Replication : 다른 리전으로의 버킷 복제
-원하는 각 리전에 설정되어 있어야 함 (전세계 대상x)
-파일 거의 실시간(near real-time)으로 갱신
-캐싱x, 읽기 전용으로만 설정 가능
-일부 리전을 대상으로 동적 컨텐츠를 낮은 지연시간으로 제공하고자 할 때 유용

6) ALB or EC2 as an origin

-사용자 지정 HTTP 백엔드에 접근(ALB, EC2 인스턴스 포함)



- EC2 인스턴스 통해 HTTP 백엔드를 개발할 시
CloudFront 통해 접근하길 원함
→CloudFront의 Edge Location에 접근
→app이 EC2 인스턴스에 요청 보냄
- all Public 설정
(그렇지 않으면 CloudFront에는 가상 프라이빗
클라우드가 x기 때문에 인스턴스에 접근x)
- Edge Location의 모든 Public IP가 EC2에 접근
가능하도록 SG 설정



- ALB ⇒ Public 설정
- 백엔드 EC2 인스턴스 ⇒ Private 설정 가능
(∵로드밸런서와 인스턴스 간에 가상 프라이빗 net
설정되어 있어서)
- EC2 인스턴스 SG ⇒ 로드밸런서 허용하도록 설정
- app Public IP가 로드밸런서 SG 정책에 허용되어
있어야 함
(∵사용자가 엣지 로케이션으로 접근 시 연결가능
하도록)

7) Geo Restriction (지리적 제한 기능)

-사용자 지역에 따라 배포 객체 접근 제한 O
① Whitelist: 접근 가능한 국가 목록 만들어 설정
② Blacklist: 접근 불가능한 국가 목록 만들어 설정
-여기서 '국가'는 3rd party 지역 DB에서 설정 → 사용자의 IP가 어떤 국가에 해당하는지 확인 가능
-Use case: 컨텐츠 저작권법으로 인한 제한

8) 가격

-Edge Location마다 데이터 전송 비용 ≠

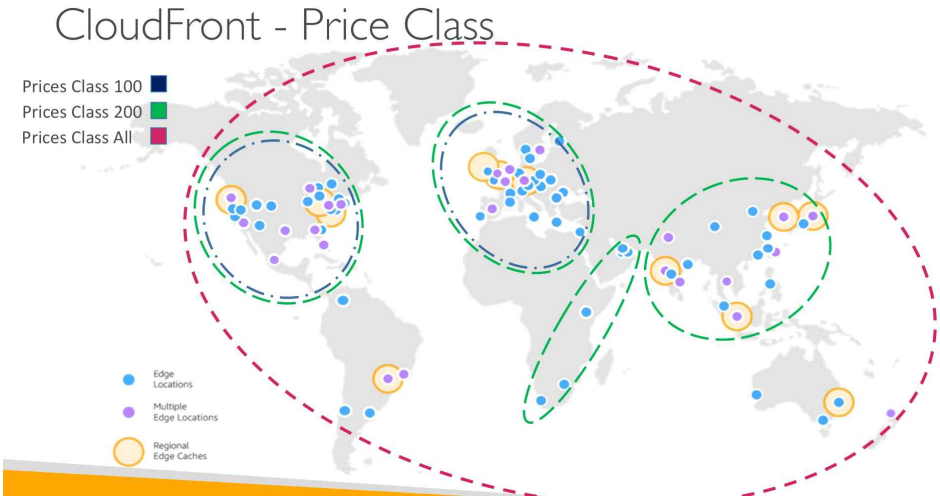
-CloudFront에서 더 多데이터 전송될수록 → 비용⇓

9) Price Classes (가격 등급)

-비용절감 위해 CloudFront 분산할 전세계 Edge Location ⇓이는 방법 있음

-가격 등급 종류

① Price Class All	all 리전 사용, 최상의 성능 제공, 비용 다소 多
② Price Class 200	대부분 리전 사용 가능, 가장 비싼 리전 제외
③ Price Class 100	가장 저렴한 리전만 사용



10) Cache Invalidations (캐시 무효화)

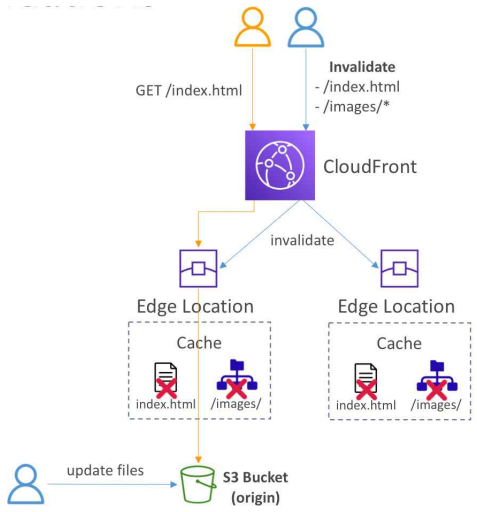
-백엔드 오리진을 업데이트 하면 TTL로 인해 업데이트된 콘텐츠 받을 수 x

∴새 콘텐츠를 빨리 받고 싶을 때 사용

-전체 or 일부 캐시 강제 새로고침 → 캐시에 있는 TTL 모두 제거 0

이를 위해 캐시 무효화 실행해야 함

*TTL (Time To Live)
:키가 만료될 때까지의 시간을 지정



- 특정 파일 경로로 전달해야 함
  ① 특정 파일 무효화
  ② 특정 경로 무효화
  → 엣지 로케이션의 캐시에 있는 all 이미지 지움

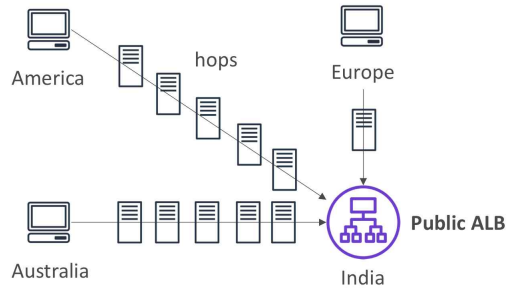
다음은 몇 가지 예입니다:

- 특정 디렉터리의 모든 파일을 무효화하려면
  `/directory-path/*`
- 특정 디렉터리와 그 하위 디렉터리 전체, 디렉터리와 하위 디렉터리에 들어 있는 모든 파일을 무효화하려면
  `/directory-path*`
- logo.jpg, logo.png, log.gif와 같이 같은 이름에 다른 파일 이름 확장명을 지닌 모든 파일을 무효화하려면
  `/directory-path/file-name.*`
- 특정 디렉터리의 전체 파일 중 파일 이름이 같은 문자로 시작하는 파일(예: HLS 포맷 비디오용 파일 전체)을 무효화하려면
  `/directory-path/initial-characters-in-file-name*`
- 쿼리 문자열 파라미터에 따라 캐싱하도록 CloudFront를 구성할 때 각 버전의 파일을 무효화하려는 경우
  `/directory-path/file-name.file-name-extension*`
- 특정 배포의 모든 파일을 무효화하려면
  `/*`

https://docs.aws.amazon.com/ko_kr/AmazonCloudFront/latest/DeveloperGuide/Invalidation.html

2. AWS Global Accelerator

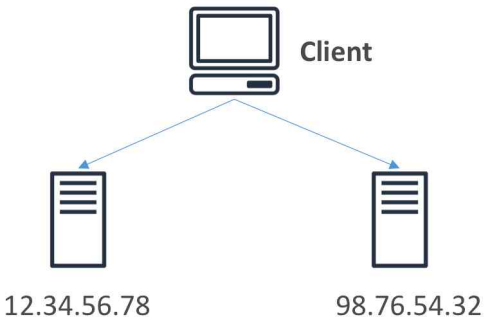
1) Global users for our app



- 사용자들이 app에 접근 시 공용 인터넷 사용하는데 라우터 거치는 동안 수많은 홉(라우터 or 서버 등)으로 인해 지연 발생 가능

⇒ Global Accelerator 사용

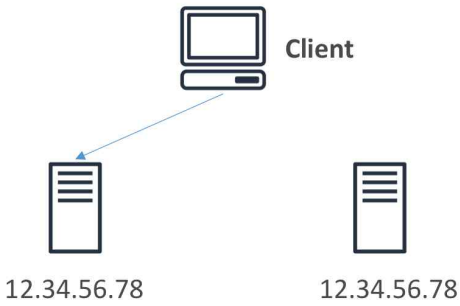
2) Unicast IP vs Anycast IP



<Unicast IP>

- 하나의 서버가 하나의 IP 주소 가짐

⇒ 12로 시작하는 IP 주소를 참조한다면 왼쪽 서버로 연결



<Anycast IP>

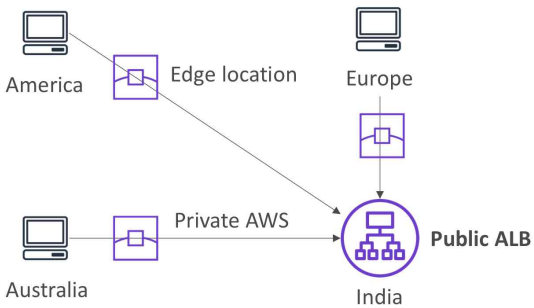
- 모든 서버가 동일한 IP 주속 가짐

⇒ 클라이언트는 가장 가까운 서버로 라우팅 됨

3) AWS Global Accelerator

-app을 위해 2개의 Anycast IP 사용

-app 라우팅하기 위해 AWS 글로벌 net 활용



- Anycast IP ⇒ 사용자와 가장 가까운 Edge location으로 트래픽 직접 전송
- 더 안정적, 지연시간이 적은 Private AWS net 거쳐 ALB로 트래픽 전송
- ex) 사용자들이 전세계에 걸쳐 있는데 미국에서 인도로 라우팅 하고 싶을 때

  ⇒미국의 공용 인터넷 거쳐서 보내는 대신 가장 가까운 엣지 로케이션과 통신함.

  엣지 로케이션부터 내부 AWS net 거쳐 ALB로 연결

-Elastic IP, EC2 인스턴스, ALB, NLB와 함께 작업

-public or private일 수 0

-net 거치기 때문에 안정적 성능 보여줌

- 지능형 라우팅으로 지연시간이 가장 짧은 엣지 로케이션으로 연결됨, 잘못될 경우 신속한 리전 장애 조치
- 2개의 Anycast IP 변하지x → 클라이언트 캐시에 문제 x
- 내부 AWS net 거쳐 안정적

-Health Checks

• Global Accelerator ⇒ app에 대해 상태확인 할 것임
• app이 글로벌한지 확인(한 리전에 있는 한 ALB에 대해 상태확인 실패 시 → 1분안에 자동화된 장애조치)
• 재해복구에 뛰어남

-보안
• 클라이언트가 Whitelist해야하는 단 2개의 외부 IP만 존재 → 안전
• DDoS 보호 자동으로 받음(∵AWS Shield 덕분에)

3. CloudFront vs AWS Global Accelerator

CloudFront	AWS Global Accelerator
- 글로벌 net 사용 - AWS가 생성한 전세계의 엣지 로케이션 사용 - DDoS 보호 위해 AWS Shield와 통합	
• 이미지 or 비디오 같은 캐시 가능한 내용 & API 가속 및 동적 사이트 전달 같은 동적 내용 all에 대해 성능 ⇑ • 캐시된 내용을 엣지로부터 가져와 전달	• TCP or UDP 상의 다양한 app 성능 ⇑ • 패킷 ⇒ 엣지 로케이션으로부터 하나 이상의 AWS 리전에서 실행되는 app으로 프록시 됨 (캐싱 불가) • Use case -게임, IoT, Voice over IP 같은 비 HTTP 사용 시 -글로벌하게 고정 IP 요구하는 HTTP 사용 시 -결정적이고 신속한 리전 장애 조치 필요 시