# SI에서 배운 AWS 안티패턴 🔥

실무에서 자주 만나는 안티패턴 5가지와
이를 해결할 수 있는 방법들

2023. 04. 19 - @kim_na_bi_

# 만나서 반갑습니다!

## 저는 나비 라고 합니다 🐈

- (전) SCVSoft 테크리드
- 3년차 ~~프론트엔드~~ 웹 풀스택 개발자
- 프론트엔드, 백엔드, DevOps, SRE 까지 모든 업무를 맡아서 함

# 이 발표에서 다루는 것

- AWS 안티패턴 다섯가지를 소개합니다.
- 안티패턴을 해결하는 방법도 소개합니다.

# 이 발표를 듣는 방법

- 🌶️ = 위험도*
- 첫번째 슬라이드에서는 안티패턴 사례**를 설명합니다.
- 두번째 슬라이드에서는 일어날수 있는 문제점을 설명합니다.
- 마지막 슬라이드에서는 예방하는 방법을 설명합니다.

* 위험도는 실제 사고가 일어났을 때 미칠 수 있는 영향을 개인적으로 상상하여 설정했습니다.
** 위 사례는 모두 픽션이며 등장하는 인물, 회사, 사건 등은 실존하는 것과 일체 관계가 없습니다.

# S3를 CDN 없이 정적 웹 호스팅으로 사용하는 경우

- 성수에 있는 게임회사에서 일하는 철수는 신규로 런칭하는 게임의 소개 홈페이지를 S3에 직접 배포하기로 마음 먹었습니다.
- 한국 리전에 버킷을 만들고, 모든 자료를 S3 버킷에 집어넣었습니다.
- 철수가 만든 홈페이지는 한국에서는 빠르게 접속되었지만, 미국에서는 느리게 접속되는 문제가 있었습니다.
- 또, 트래픽이 많이 몰리는 날에는 홈페이지에 비용이 많이 청구되는 문제도 생겼습니다.
- 조금 더 나은 방법이 있지 않았을까요?

# S3를 CDN 없이 정적 웹 호스팅으로 사용하는 경우

- S3 버킷은 static hosting을 하기에 적합한 환경이 아닙니다.
  - S3 버킷은 한 리전에 위치하여 있어 유저가 버킷이 위치한 리전과 먼 거리에 위치할 경우 느린 응답을 받을 수 있습니다.
  - S3는 HTTPS 연결을 지원하지 않아 안전한 연결이 불가능합니다.
  - S3는 캐싱이 되지 않아 요청이 발생할때마다 객체를 일일이 불러오게 되어 트래픽이 많아질수록 비용이 많이 발생하게 됩니다.

## S3를 CDN 없이 정적 웹 호스팅으로 사용하는 경우

- CloudFront를 애용합시다.
  - 비용이 훨씬 저렴합니다.
  - 버튼 몇번이면 권한 연결도 간단합니다.
- https://aws.amazon.com/ko/premiumsupport/knowledge-center/cloudfront-access-to-amazon-s3/

# RDS를 백업하지 않고 사용하는 경우

- 철수의 회사는 개발용, 프로덕션용 RDS를 운영하고 있습니다.
- 여러 이유로, 각 RDS의 자동 백업은 꺼져있었습니다.
- 바로 그때 철수는 개발 서버에 돌려야 하는 DB 초기화 코드를 프로덕션에 적용하는 사소하지만 중대한 실수를 해버렸습니다.
- 실제 유저의 데이터가 모두 사라졌고, 운영팀에서는 공지를 내보내며 업무를 수습하기 시작했습니다.
- 과연 어떻게 하면 이 문제를 막을 수 있었을까요?

# RDS를 백업하지 않고 사용하는 경우

- RDS를 백업하지 않고 사용하는 경우 철수와 같이 사소하지만 중대한 실수가 발생했을 때 유저의 데이터를 모두 날리는 끔찍한 결과가 발생할 수 있습니다.
- 프로덕션용 DB에 개발자가 쉽게 접근할 수 있는 환경을 만들어서는 안됩니다.

## RDS를 백업하지 않고 사용하는 경우

- 프로덕션용 RDS에 자동 백업을 꼭 켜둬야 합니다.
  - 물론 비용이 조금 더 나가겠지만, 사고가 일어났을 때 복구하는 비용보다는 저렴합니다.
- DB에 실행할 수 있는 SQL 명령어의 권한을 개발자별로 분리합니다.

## 스케일 인-아웃이 수동으로 이루어지는 경우

- 철수가 만든 게임 서버의 백엔드는 단일 EC2로 이루어져 있습니다.
- 홍보팀이 일을 열심히 해 게임은 대박이 났고, 하루 최대 접속자가 10만명이 넘는 큰 게임이 되었습니다.
- 바로 그때, 게임 서버의 백엔드가 과부화로 인해 CPU 버스트되며 게임에 접속이 안되는 이슈가 발생하기 시작했습니다.
- 과연 어떻게 하면 서비스가 다운되는것을 막을 수 있었을까요?

🌶️🌶️🌶️

# 시스템 스케일 인-아웃이 수동으로 이루어지는 경우

- EC2가 오토스케일링 그룹 없이 홀로 있는 경우 사용자가 갑자기 많아질 때 서비스가 과부화되어 버스트될 위험이 있습니다.
- 물론 사람이 직접 모니터링을 하며 ec2 인스턴스의 사이즈를 유동적으로 늘리거나 줄일수도 있겠지만, 시간 대비 효율이 부족합니다.

🌶️🌶️🌶️

## 시스템 스케일 인-아웃이 수동으로 이루어지는 경우

- 시스템 볼륨의 경우에는 Systems Manager Automation 를 사용하여 자동으로 확장하는 기능을 만들 수 있습니다.
  - https://repost.aws/ko/knowledge-center/ec2-volume-disk-space
- EC2 / RDS의 경우에는 오토스케일링 그룹을 지정함으로써 쉽게 해결할 수 있습니다.
  - https://docs.aws.amazon.com/ko_kr/autoscaling/ec2/userguide/get-started-with-ec2-auto-scaling.html

🌶️🌶️🌶️🌶️🌶️🌶️

## 다양한 서비스가 한 서버에 있는 경우

- 철수가 만든 EC2 백엔드 안에는 이번에 만든 게임서버 말고도, 이전에 출시한 게임의 운영서버나, 어드민 관제 툴과 같이 대부분의 서비스가 한 서버에 있습니다.
- 앞선 사례의 연장선으로, 철수가 이번에 런칭한 게임 서버가 인기가 많아져 과부화됨에 따라 EC2가 CPU 버스트되었고…
- 이전에 운영중이던 게임의 서버부터, CS 대응을 위한 어드민 툴까지 모두 장애가 발생하는 문제가 발생했습니다.
- 과연 어떻게 했다면 이 장애를 막을 수 있었을까요?

🌶️🌶️🌶️🌶️🌶️🌶️

## 다양한 서비스가 한 서버에 있는 경우

- 모든 서비스를 한 서버에 넣어두는 경우 다음과 같은 문제가 발생할 수 있습니다.
  - EC2가 해킹당할경우 해커가 손쉽게 서비스의 모든 정보를 가져갈 수 있습니다.
  - 공통된 자원을 사용하다보니, 한 서비스가 많은 자원을 차지할 경우 다른 서비스에 할당되는 자원이 부족해질 수 있습니다.
  - 한 서비스가 여러 이유로 다운될 경우 다른 서비스까지 다운될 가능성이 높습니다.

🌶️🌶️🌶️🌶️🌶️🌶️

## 다양한 서비스가 한 서버에 있는 경우

- 각 서비스의 기능별로 서버를 분리해야합니다.
- BeanStack / RDS / S3 / EKS / ElasticCache 같은 서비스를 사용하면 더욱 편합니다.

 x 무제한

# 🌶️ x 무제한

## ROOT 권한을 가진 IAM을 만드는 경우

- 개발자 철수는 AdministratorAccess 권한을 가진 IAM을 만들었습니다.
- 이번에 새로 런칭하는 GameLift 기반 서비스에 접속하기 위해 철수는 방금 만든 IAM의 엑세스 키와 시크릿 키를 코드에 평문으로 입력하고, 빌드했습니다.
- 철수의 게임은 성공적으로 런칭했고, 이용자도 많이 접속했습니다.
- 그리고 AWS 계정에 해킹이 발생하여 1억 7천만원이 넘는 금액이 청구되었습니다.
- 과연 무엇이 문제였을까요?

# 🌶️ x 무제한

## ROOT 권한을 가진 IAM을 만드는 경우

- AdministratorAccess 권한이 있는 IAM이 외부에 노출될 경우 해커는 다양한 방법을 통해 AWS 리소스를 해킹할 수 있습니다.
- 또, 클라이언트에 평문으로 IAM 키를 저장하는 경우 파일 디코더등을 사용하여 역분석을 하는 경우 쉽게 시크릿 키가 노출이 될 수 있습니다.

# 🌶️ x 무제한

## ROOT 권한을 가진 IAM을 만드는 경우

- **AdministratorAccess 권한을 가진 IAM을 만들지 않는다.**
  - 부득이하게 만들어야 할 경우, 제한된 사람에게만 공유한다.
- **.env 파일에 IAM의 엑세스 키를 포함하지 않는다**
  - 대신 AWS IAM Role을 사용한다.
  - https://www.nabi.kim/aws-iam-role 
- 각 서비스마다 분리된 권한을 가진 IAM을 만든다.
  - https://policysim.aws.amazon.com 에서 정책이 잘 적용되었나 테스트할 수 있습니다.

# Q & A 

감사합니다 🔥