PORTFOLIO

AWS 인프라의 진화: EC2, K8s, 그리고 EKS 아키텍처 구축

SNS

fw4568

Phone

010-8278-3213

Email

fw4568@gmail.com

INTRODUCTION

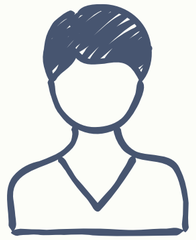
안녕하세요, 오지민입니다.

"인프라가 멈추면, 비즈니스도 멈춥니다."

01. 경험(Experience)
물류 현장에서 근무하며 PDA 접속 장애가 전체 공정을 멈추는 것을 목격했습니다.
1분 1초가 급한 현장에서 , 안정적인 서버와 네트워크가 비즈니스의 핵심이라는 것을 느꼈습니다.

02. 역량 (Competence)
이 경험을 바탕으로 클라우드 엔지니어로 전향했습니다. 단순 구축을 넘어 AWS-Azure 멀티 클라우드 DR 환경을 직접 설계하며, 어떤 상황에서도 서비스가 중단되지 않는 고가용성 아키텍처를 구현할 수 있습니다.

03. 목표 (Vision)
입사 후 기존 인프라를 빠르게 파악하여 운영 안정성에 기여하겠습니다. 장기적으로는 비즈니스 연속성을 책임지는 클라우드 아키텍트로 성장하겠습니다.



Oh-jimin Profile

교육

AI MSP 멀티클라우드 엔지니어 부트 캠프
2025.07-2026-01

- **멀티 클라우드 DR** : 멀티 클라우드 간 DR 구현
- **고 가용성 아키텍처 설계** : Karpenter + HPA + ToplogySpeard 노드/AZ 고가용성
- **IaC: Terraform** 활용한 EKS 인프라 배포 자동화 구현
- **CICD Pipeline**: CI/CD 파이프라인(Jenkins, ArgoCD,Canary 배포) 운영
- **모니터링**:오토스케일링 및 모니터링(Prometheus/Grafana) 대시보드 구축

경력

대한 통운 2CP
2020.03 ~ 2024.03

넷코아 테크
2018.01 ~ 2018.12

맨파워 물류
2017.01 ~ 2017.11

자격증

AWS SAA

정보보안 기사 (필기)

리눅스 마스터 1급

컴퓨터 활용 능력 1급

PROJECTS

1. 인스턴스 환경
2. 인스턴스 K8S
3. EKS 환경

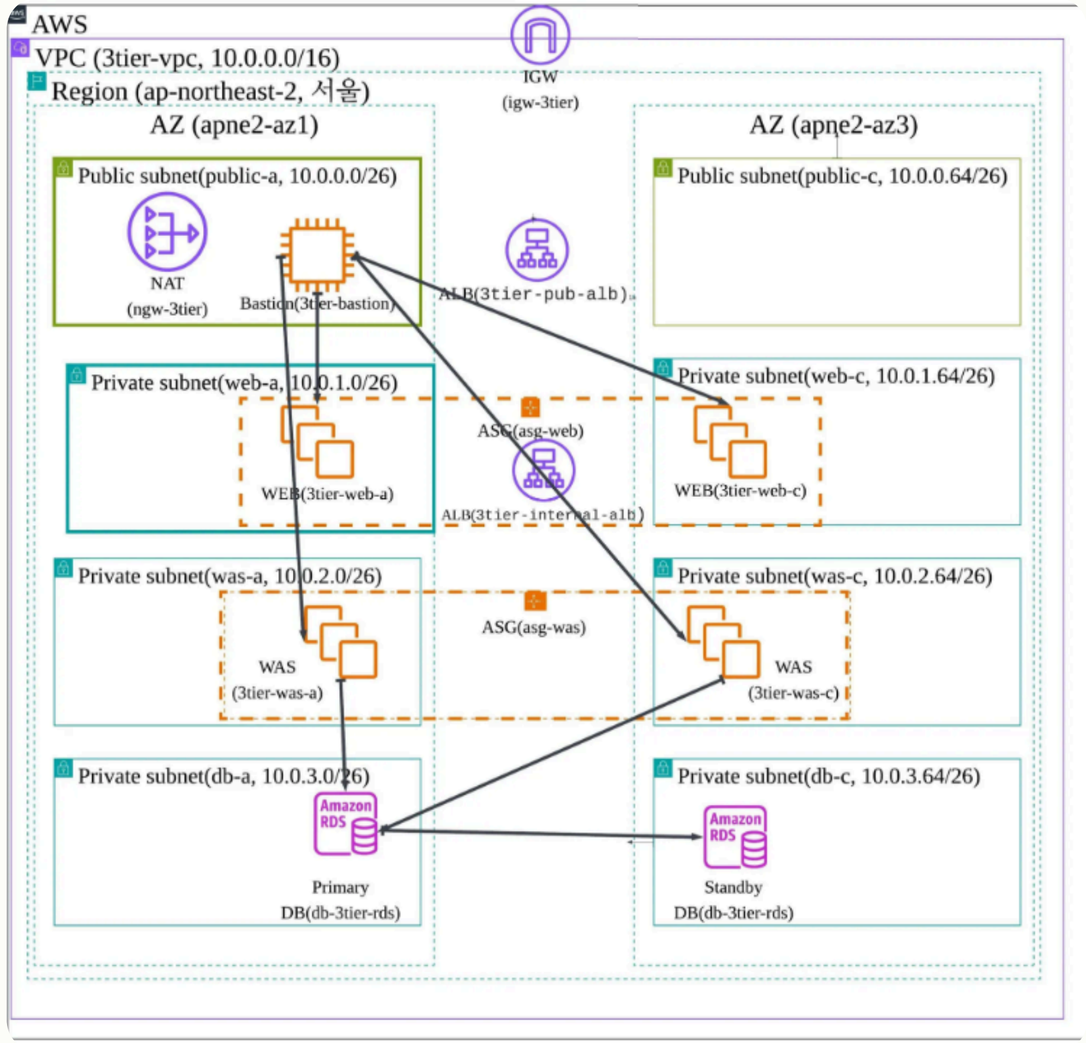
PROJECTS 1 인스턴스 환경

AWS 인스턴스 3tier 구축

- **Network & Security**
  - **VPC 설계**: 두 가용영역을 사용해 고 가용성을 가짐
  - **Subnet 계층화** : Public,Private-web,Private-was,Private-DB 4계층으로 서브넷을 철저히 분리
  - **보안 접근**: 베스천을 통해 프라이빗 자원이 접근 가능
  - **SG 분리** : 계층별로 분리를 통해 보안 및 관리가 효율 증대

- **LB & HA**
  - Public ALB를 통해 들어온 트래픽을 WEB 으로 분산
  - WEB 과 WAS 사이에 Internal ALB를 배치 (Proxy path)
  - Auto scaling 트래픽 변동에 대응 하기 위해 사용
  - Multi-AZ(Primary-standby) 장애시 고가용성 보장 (다운타임 존재)

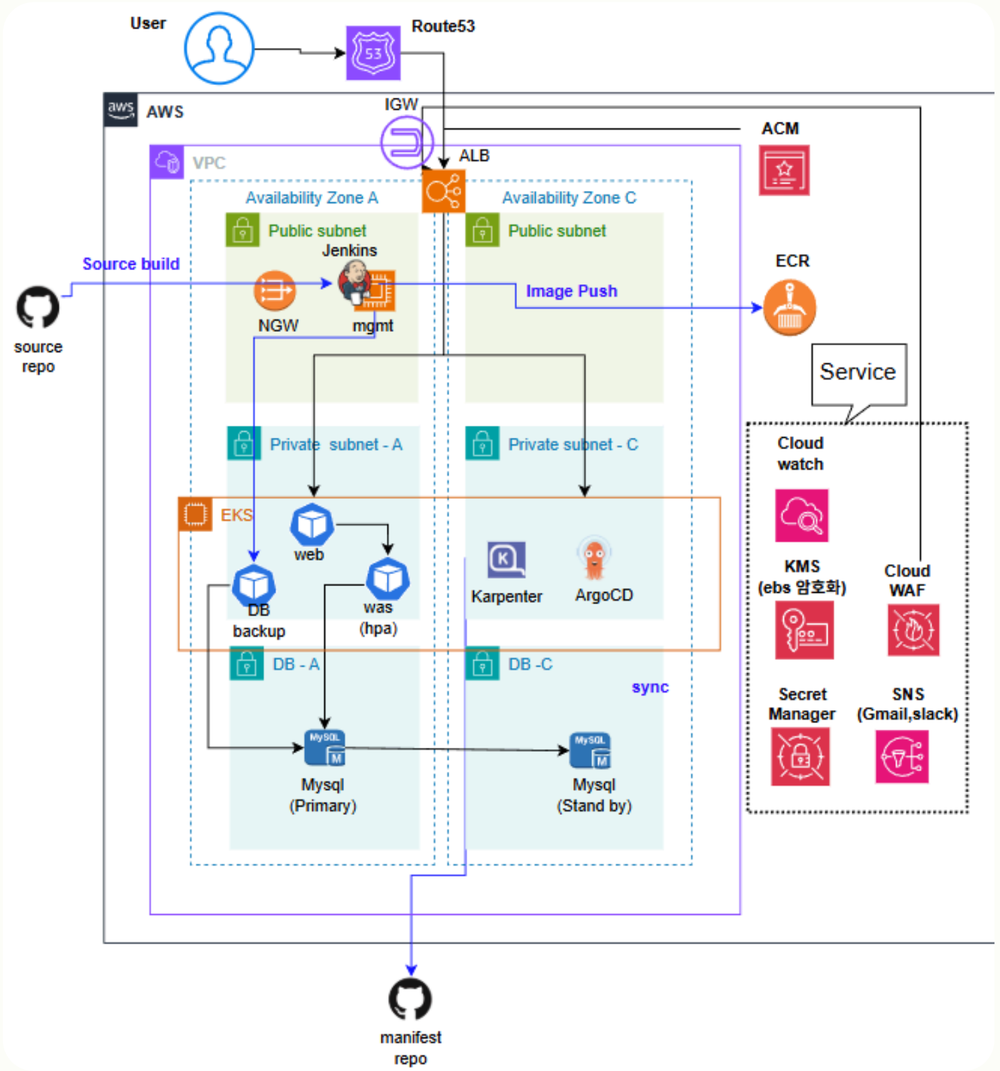
- **개선점**
  - 배포의 속도 와 리소스 효율성 측면에서 개선이 필요
  - 급격한 트래픽(k6 부하테스트) **대응 속도가 느려** **컨테이너 기반 환경으로 마이그레이션** 결정



PROJECTS 2 인스턴스 K8S

AWS 인스턴스 K8s 구축

- **아키텍처**
  - Contorol Plane & Worker Node 물리적 분리
  - **Network(CNI)**: Clium (eBPF) 도입으로 iptables 성능한계 극복 및 가시성 확보
  - **Version**: Kubernetes v1.31
- **고가용성**
  - **Auto Scaling**: Karpenter + HPA 이중 구성
  - **Ingress**: Nginx Proxy 설정을 K8s Ingress로 전환 L7 라우팅 관리
- **CI/CD 파이프라인**
  - Jenkins & ArgoCD를 활용한 GitOps 기반 배포 자동화



- **개선점**
  - **모니터링 시스템 부재 및 알람**
  - **AZ 장애 시 WEB/WAS 자가복구 불가능**
  - **AWS 장애시 대책이 존재 하지 않음**
  - **세션 관리 부재**

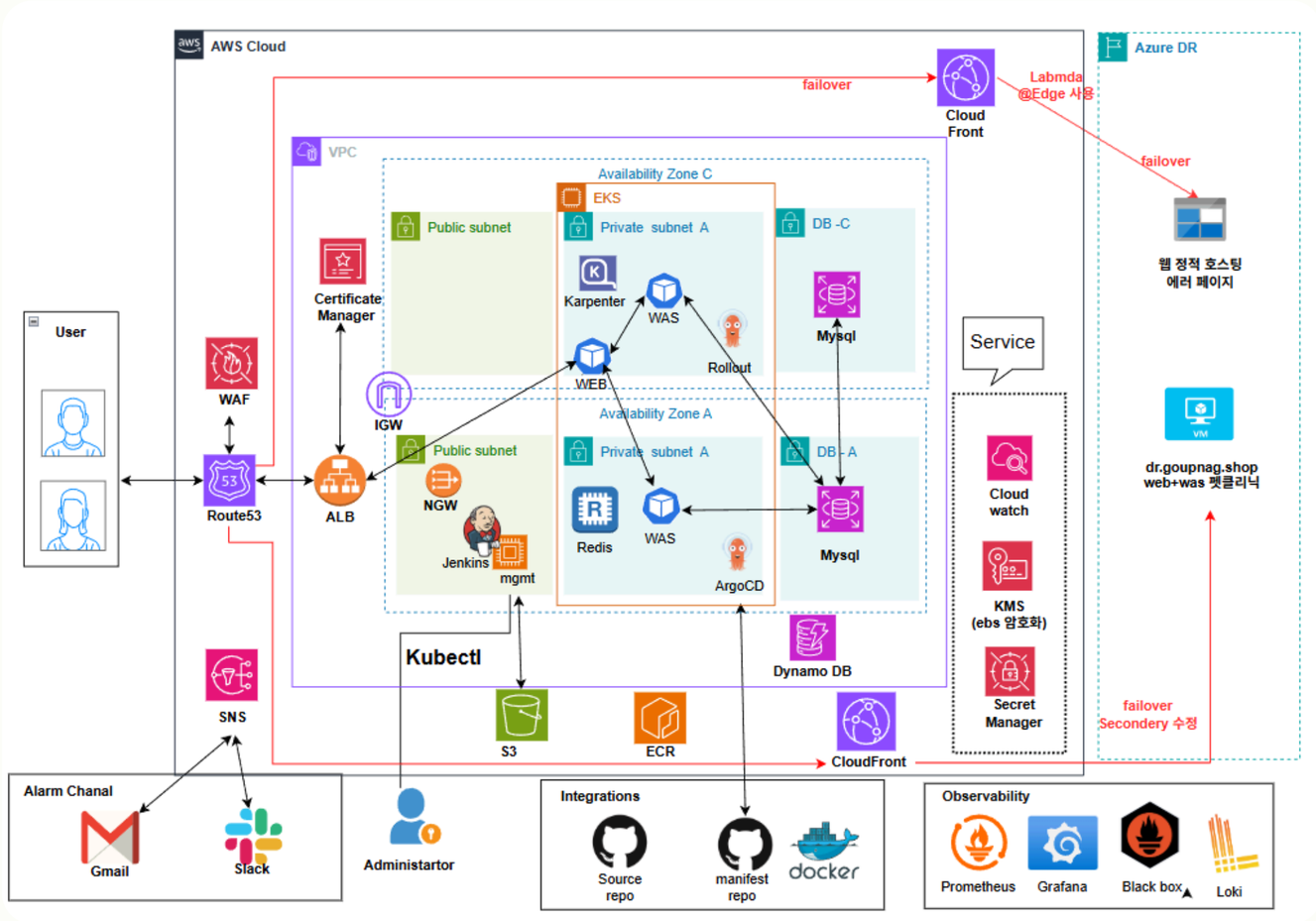
트러블 슈팅 및 정리 페이지

PROJECTS 3 EKS 환경

EKS를 활용한 아키텍처 구축 및 운영

- **Session Architecture (데이터 정합성)**
  - **Redis Clustering**: ElastiCache(Redis)를 도입하여 배포 시 세션 끊김 방지 및 정합성 확보
  - **Multi-AZ Sync**: 가용 영역(AZ) 간 데이터 동기화를 통해 장애 상황에서도 서비스 연속성 유지

- **Observability (모니터링 & 로깅)**
  - **PLG Stack**: **Loki + Promtail + Grafana**를 구축하여 컨테이너 로그 중앙 집중화 및 시각화
  - **Alert Pipeline**: **Alertmanager**를 통해 Karpenter 스케일링, Jenkins 빌드 오류, 백업 누락 시 슬랙 알림

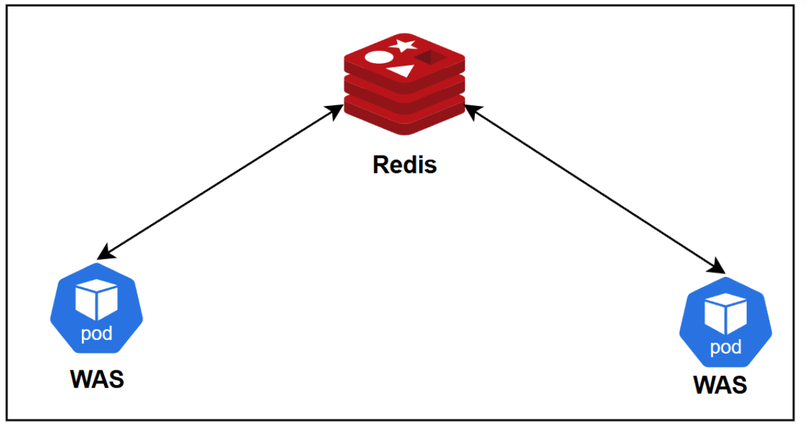


자세한 내용 → [프로젝트 블로그 정리]

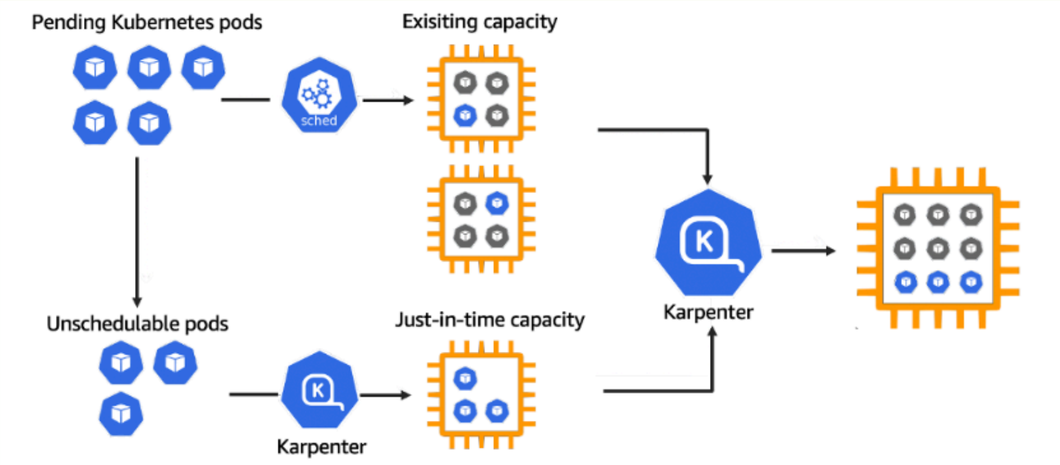
PROJECTS 3 EKS 환경

아키텍처 구성 상세

- **배포 과정에서 세션 이슈로 Redis 구축**
  - 로그인 시도시 다른 pod 에서는 **세션이 없어 인증이 실패**
  - 세션 아키텍처 사용
  - 운영 오버헤드가 낮은 **Redis** 를 선택



- **Kapenter 및 스케줄링 정책을 통한 고가용성 확보**
  - **Karpenter + HPA + EKS** 구성
  - AZ 장애 대비 하여 **Topology Spread Constraints** / **Max skew** 설정
  - 안정적인 운영작업을 위한 **PDB** 설정



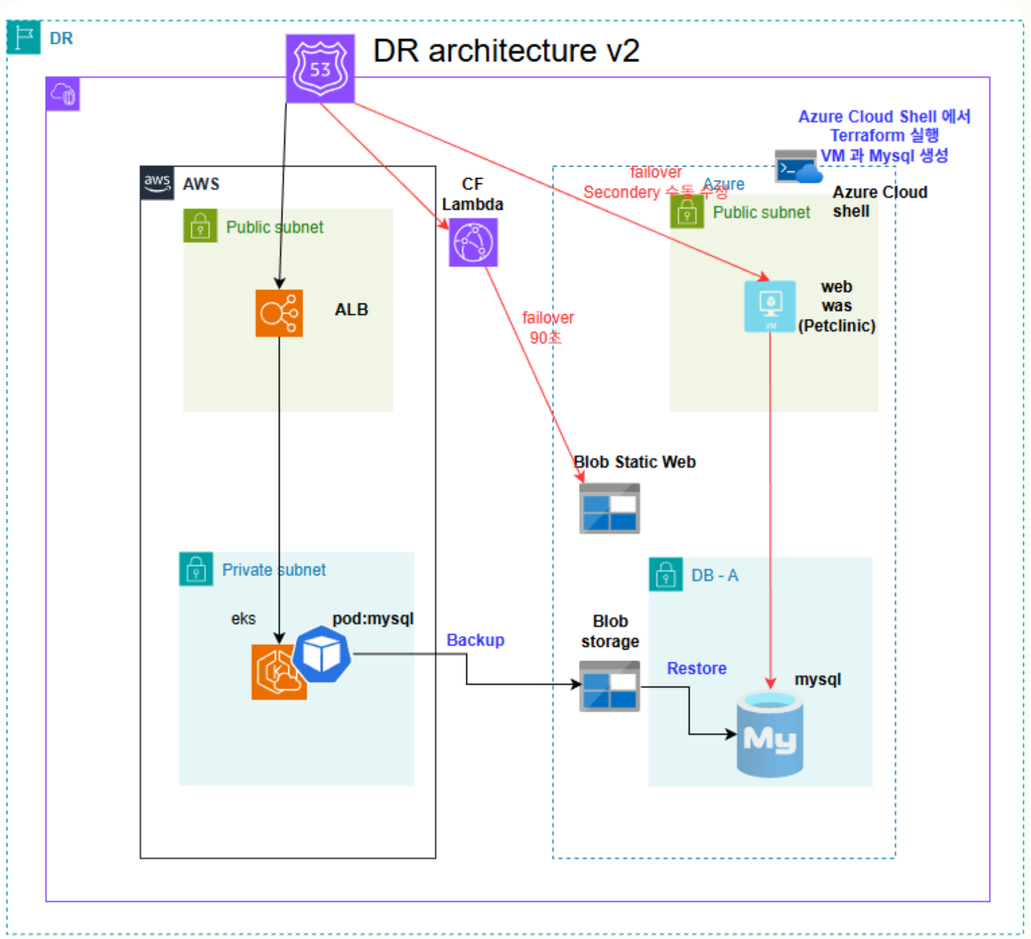
PROJECTS 3 EKS 환경

멀티 클라우드 DR 아키텍처

- **재해 복구전략**
  - **비용 최적화**: **backup/restore**으로 관리하는 **Cold Standby** 전략 채택
  - **RPO/RTO**: 백업 주기와 복구 자동화를 통해 **RPO 24시간 / RTO 30분 목표 달성**

- **DR System (AWS-Azure 하이브리드)**
  - **Fail over :** AWS 장애 시 Route53이 이를 감지하여 Azure 정적 페이지로 트래픽 즉시 우회
  - 테라폼 코드로 30분 내 **WEB/WAS 인프라 구축**

- **앞으로 개선해야할 점**
  - Blob 대신 SWA를 사용하여 정적 웹 처리를 수정 → **SWA**는 헤더 수정 가능 **웹 유연성 확보**
  - Dump 기반 복구 방식에서 나아가, **CDC 솔루션** 도입을 통해 **이기종 클라우드간 데이터 동기화 검토**

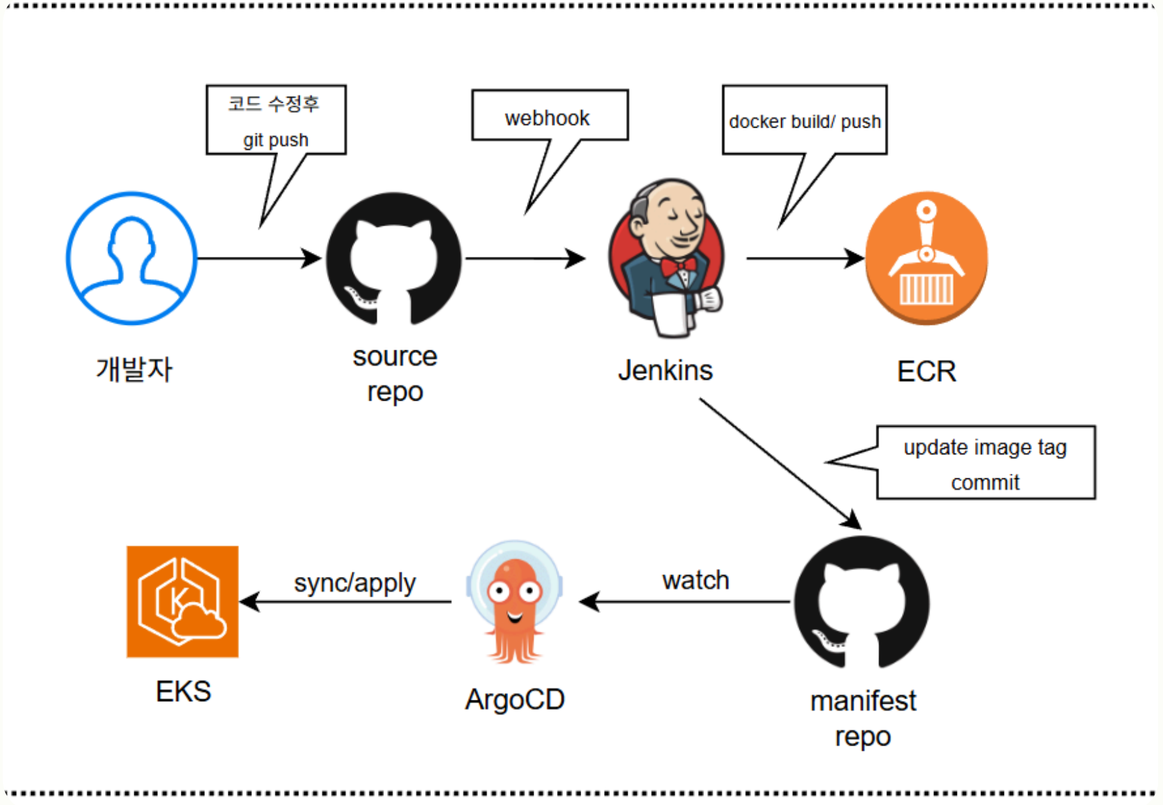


PROJECTS 3 EKS 환경

CI/CD 파이프라인

- **CI/CD 파이프라인**
  - ECR 을 통해 불변성 확보
  - GitOps 통해 인프라 및 배포 형상 관리의 **SSOT 역할** 수행
  - **Build Hash + Version:**이미지 무결성 추적성 확보
  - **ignoreDifferences** : 배포와 오토 스케일링의 역할 분리 및 충돌 방지

- **Canary 배포 방식 사용**
  - 가중치를 통한 트래픽 비율 지정
  - 정확한 트래픽 분산을 위해 ALB Traffic routing 을 사용
  - 트래픽 분산 정리 : 10% → 50% → 90% → 100% 으로 초기 오류 최소화

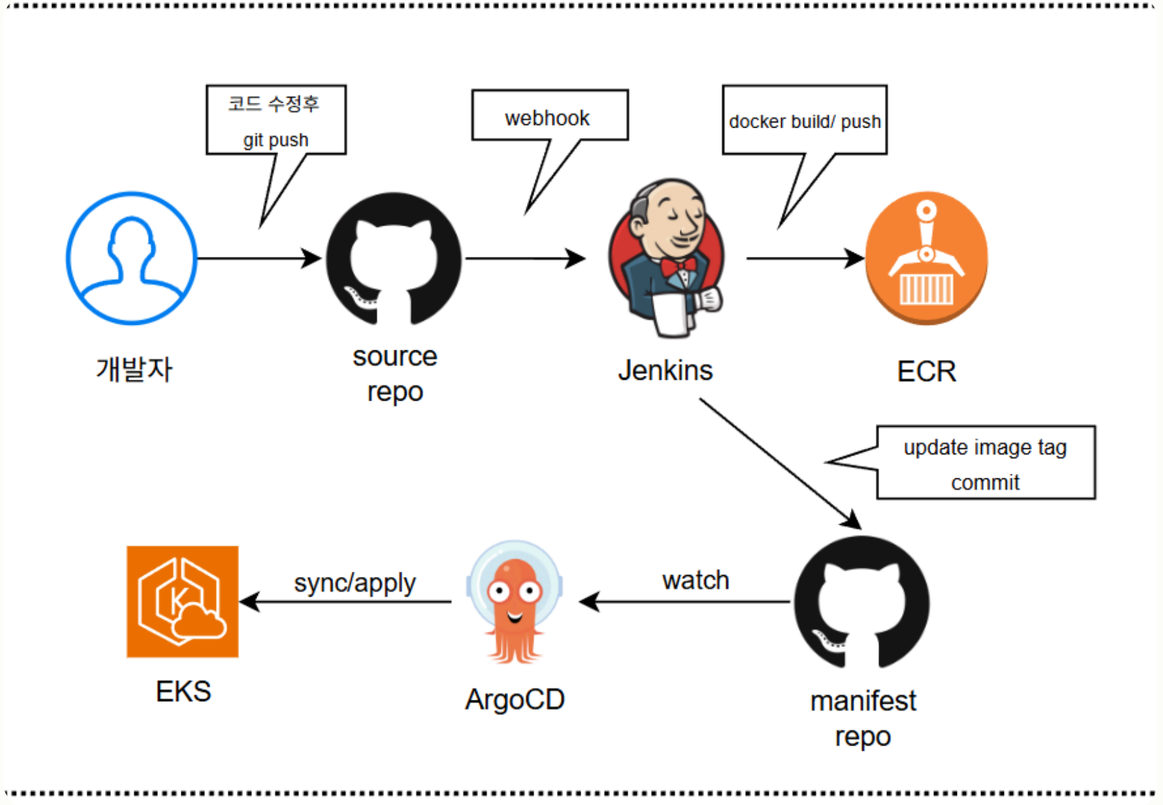


PROJECTS 3 EKS 환경

트러블 슈팅

- **CI/CD 파이프라인**
  - ECR 을 통해 불변성 확보
  - GitOps 통해 인프라 및 배포 형상 관리의 **SSOT 역할** 수행
  - **Build Hash + Version:**이미지 무결성 추적성 확보
  - **ignoreDifferences** : 배포와 오토 스케일링의 역할 분리 및 충돌 방지

- **Canary 배포 방식 사용**
  - 가중치를 통한 트래픽 비율 지정
  - 정확한 트래픽 분산을 위해 ALB Traffic routing 을 사용
  - 트래픽 분산 정리 : 10% → 50% → 90% → 100% 으로 초기 오류 최소화



감사합니다.