AWS 백서

AWS 기반 5G 네트워크를 위한 지속적 통합 및 지속적 전달

AWS 기반 5G 네트워크를 위한 지속적 통합 및 지속적 전달: AWS 백서

Table of Contents

요약 i
요약 1
소개 2
지속적 통합 및 지속적 전달 4
지속적 통합 4
지속적 전달 및 배포 4
코드형 인프라 4
AWS의 CI/CD 5
AWS 기반 5G 네트워크 8
5G 네트워크의 CI/CD 9
CI/CD 세부 단계 11
네트워크 설정 11
인프라 배포 11
클라우드 네이티브 네트워크 기능 배포 12
CFN 지속적 전달 14
보안 16
관찰 17
서드 파티 및 오픈 소스 도구를 사용한 CI/CD 오케스트레이션 20
Terraform 20
인프라 배포 20
네트워크 기능 배포 및 구성 22
테스트 24
CI/CD 및 오케스트레이션 26
결론 27
기여자 28
문서 개정 29
참고 문헌 30
약어 31
고지 사항 33

AWS 기반 5G 네트워크를 위한 지속적 통합 및 지속적 전달

게시 날짜: 2021년 3월 8일(문서 개정)

요약

이 백서에서는 5G 네트워크를 위한 지속적 통합 및 지속적 전달(CI/CD)과 Amazon Web Services(AWS) 도구 및 서비스를 사용하여 5G 네트워크 기능의 배포 및 업그레이드를 완벽하게 자동화하는 방법을 소개합니다. 그리고 네트워크 설정, 인프라 배포, 클라우드 네이티브 네트워크 기능 배포 및 네트워크 기능의 지속적인 업데이트를 포함하여 5G 네트워크 기능을 위한 CI/CD의 여러 단계를 자세히 설명합니다. 또한 테스트, 관찰 및 오케스트레이션을 위한 오픈 소스 및 서드 파티 도구와의 통합에 대한 세부 정보도 제공합니다.

이 백서는 통신 서비스 제공업체(CSP) 및 Independent Software Vendor(ISV)를 대상으로 합니다.

소개

기존에는 셀룰러 네트워크에서 새로운 네트워크 노드 또는 새로운 기능의 개발, 실험실 및 현장 통합 테스트, 프로덕션 배포는 미션/비즈니스 크리티컬 통신 서비스의 안정성을 보장하는 데 몇 주에서 길게는 몇 달이 걸렸습니다. 긴 배포 주기는 기존 네트워크 노드의 모놀리식 아키텍처, 다중 공급 업체 환경, 2G, 3G 및 4G 모바일 네트워크의 네트워크 엔터티 간 많은 지점 간 인터페이스로 인해 발생했습니다.

5G Network Evolution with AWS(AWS를 기반으로 한 5G 네트워크 진화) 백서에 소개된 바와 같이 3GPP로 표준화된 5G 네트워크는 이제 가상화 및 컨테이너화를 통해 지원되는 클라우드 네이티브 아키텍처를 지원합니다. 구체적으로 설명하면, 5G 네트워크는 마이크로 서비스, 무상태 및 서비스 기반 아키텍처의 새로운 패러다임을 도입하고 지원합니다.

이 5G 아키텍처는 서로 다른 네트워크 기능이 잘 정의된 인터페이스와 API를 통해 서로 통신하는 느슨하게 결합된 독립 서비스로 작동할 수 있음을 의미합니다. 가장 중요한 점은 각 네트워크 기능을 독립적으로 업데이트할 수 있다는 것입니다. 5G의 이러한 아키텍처 전환으로 CSP는 자동화를 통해 테스트, 보안 요구 사항 및 표준을 유지하면서 네트워크 기능에 대한 업데이트를 더 자주 롤아웃할 수 있어 민첩성과 운영 효율성을 높일 수 있습니다.

CSP를 위한 새로운 기능의 통합 및 배포는 일반적으로 네트워크 기능 공급 업체가 새로운 네트워크 기능 소프트웨어 패키지(예: 컨테이너 기반 네트워크 함수의 Docker 이미지) 또는 새로운 구성 파일(예: Kubernetes 애플리케이션 케이스의 Helm 차트)을 릴리스할 때 시작됩니다. Helm 차트는 관련된 Kubernetes 리소스 세트를 설명하는 파일 모음입니다.

5G 네트워크 기능 배포를 위해 CI/CD의 패러다임을 사용한다는 아이디어가 주목받고 있지만, 통신 산업에서 이 아이디어를 실현하는 것은 어려운 과제였습니다.

AWS는 소프트웨어 전달을 위한 새로운 CI/CD 도구 개발을 선도적으로 이끌어 광범위한 업계가 시스템 안정성과 보안을 유지하면서 소프트웨어 변경 사항을 신속하게 개발하고 롤아웃할 수 있도록 지원합니다. 이러한 도구에는 AWS CodeStar, CodeCommit, CodePipeline, CodeBuild 및 CodeDeploy와 같은 소프트웨어 개발 및 운영(DevOps) 서비스 세트가 포함됩니다.

또한 AWS는 AWS Cloud Development Kit(AWS CDK), AWS CloudFormation 및 API 기반 서드 파티 도구(예: Terraform)를 사용하여 코드형 인프라(IaC)라는 아이디어를 전파하고 있습니다. AWS는 이러한 도구를 사용하여 네트워크 기능의 배포 프로세스를 AWS 내에 소스 코드로 저장하고 CI/CD 파이프라인에서 이 IaC 소스 코드를 유지 관리하여 지속적 전달을 실현할 수 있습니다.

이 백서에서는 5G 네트워크 기능의 배포 및 업데이트를 위해 AWS IaC 및 CI/CD 도구를 활용하는 자세한 프로세스를 설명합니다. 또한 테스트, 관찰 및 오케스트레이션을 위한 서드 파티 도구와의 통합에 대해서도 다룹니다.

AWS CI/CD 도구는 5G 네트워크 기능으로 제한되지 않습니다. AWS CI/CD 도구는 4G 네트워크 배포를 자동화하는 데에도 사용되어 CSP가 4G 네트워크 기능을 빠르고 효율적으로 배포 및 업데이트할 수 있도록 합니다. 대부분의 4G 네트워크 기능은 가상 네트워크 기능(VNF)을 기반으로 합니다. AWS CloudFormation과 같은 AWS CI/CD 도구 세트를 사용하여 4G VNF의 배포를 자동화하여 4G 네트워크 배포의 규모와 시간 효율성을 높일 수 있습니다.

지속적 통합 및 지속적 전달

지속적 통합

지속적 통합(CI)은 개발자가 정기적으로 코드를 AWS CodeCommit 또는GitHub와 같은 중앙 리포지토리에 푸시하는 소프트웨어 프로세스입니다. 코드가 푸시될 때마다 자동화된 구축이 트리거되고 그런 다음 테스트가 실행됩니다. CI의 주요 목표는 초기 단계에서 코드 문제를 발견하고, 코드 품질을 개선하며, 새로운 소프트웨어 업데이트를 검증 및 릴리스하는 데 걸리는 시간을 줄이는 것입니다.

지속적 전달 및 배포

지속적 전달(CD)은 테스트 환경, 스테이징 환경 및 프로덕션 환경에 아티팩트를 배포하는 소프트웨어 프로세스입니다. 지속적 전달은 완전히 자동화하거나 중요한 시점에 승인 단계를 거치도록 설정할 수 있습니다. 이렇게 하면 필요한 모든 승인(예: 릴리스 관리 승인)이 배포 전에 제대로 이루어지도록 할 수 있습니다. 지속적 전달이 올바르게 구현되면, 개발자는 언제든지 즉시 배포가 가능하며 표준화된 테스트 프로세스를 통과한 구축 아티팩트를 보유하게 됩니다.

지속적 배포의 경우, 수정 버전이 개발자의 명시적 승인 없이 자동으로 프로덕션 환경에 배포됩니다. 즉, 이를 통해 전체 소프트웨어 릴리스 프로세스가 자동화됩니다. 따라서 제품 수명 주기 초기에 지속적인 고객 피드백 루프를 제공할 수 있습니다.

지속적 배포를 사용하면 커밋되고 자동화된 테스트를 통과한 모든 변경 사항이 자동으로 프로덕션으로 릴리스됩니다. 지속적 전달은 커밋되고 자동화된 테스트를 통과한 모든 변경 사항을 프로덕션에 즉시 전달하는 것이 목적이 아니라 모든 변경 사항이 프로덕션에 적용할 준비가 되도록 만들기 위한 것입니다.

코드형 인프라

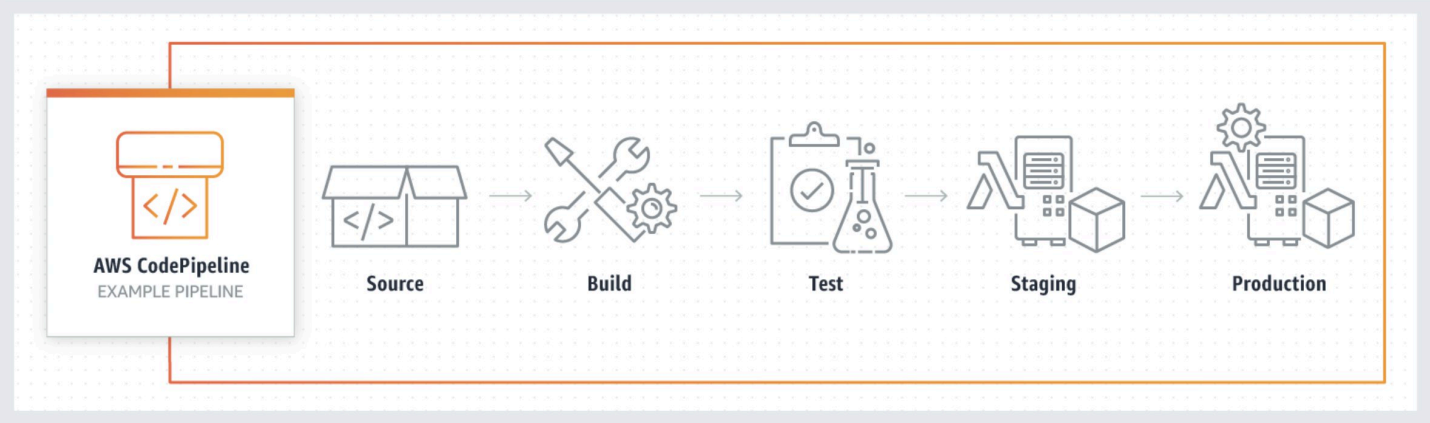
5G Network Evolution with AWS(AWS를 기반으로 한 5G 네트워크 진화) 백서에 자세히 설명되어 있듯이 IaC는 애플리케이션과 해당 환경 모두에 대한 프로비저닝 프로세스와 수명 주기 관리를 자동화하는 핵심 요인입니다. 수작업으로 수행하는 단계에 의존하지 않고 네트워크/IT 관리자와 개발자 모두 구성 파일을 사용하여 인프라를 인스턴스화할 수 있습니다. IaC는 이러한 구성 파일을 소프트웨어 코드로 취급합니다. 이러한 파일은 일련의 아티팩트, 즉 운영 환경을 구성하는 컴퓨팅, 스토리지, 네트워크 및 애플리케이션 서비스를 생성하는 데 사용할 수 있습니다. IaC는 자동화를 통해 구성 드리프트를 제거하여 인프라 배포의 속도와 민첩성을 높입니다.

AWS에서 네트워크 기능 가상화(NFV)를 구현하는 경우 이 IaC 프레임워크는 오케스트레이션 관점에서 가치를 제공합니다. Virtual Private Cloud(VPC) 생성부터 네트워크 기능 배포에 이르기까지 모든 단계를 프로그래밍하고 소스 코드로 관리하며 AWS CodeCommit에서 버전 관리를 통해 유지 관리할 수 있습니다.

네트워크 기능을 위한 이 IaC 프레임워크는 반복 가능하고 안정적인 인프라 및 네트워크 기능의 생성 및 배포를 가능하게 하며, 이는 네트워크 슬라이스 관리 및 서비스 수명 주기 관리의 엔드 투 엔드(E2E) 자동화로 확장될 수 있습니다. AWS는 AWS CloudFormation, AWS CDK, AWS CDK for Kubernetes 및 모든 AWS 서비스의 API 노출과 같은 서비스를 사용하여 프로그래밍 방식, 설명적, 선언적 방식으로 인프라를 생성, 유지 관리 및 배포할 수 있는 포괄적인 도구 세트를 제공합니다.

AWS의 CI/CD

CI/CD는 새 코드가 한쪽 끝에서 제출되고 일련의 단계(소스, 구축, 테스트, 스테이징 및 프로덕션)에 걸쳐 테스트된 후 프로덕션에서 즉시 사용 가능한 코드로 게시되는 파이프라인으로 나타낼 수 있습니다.



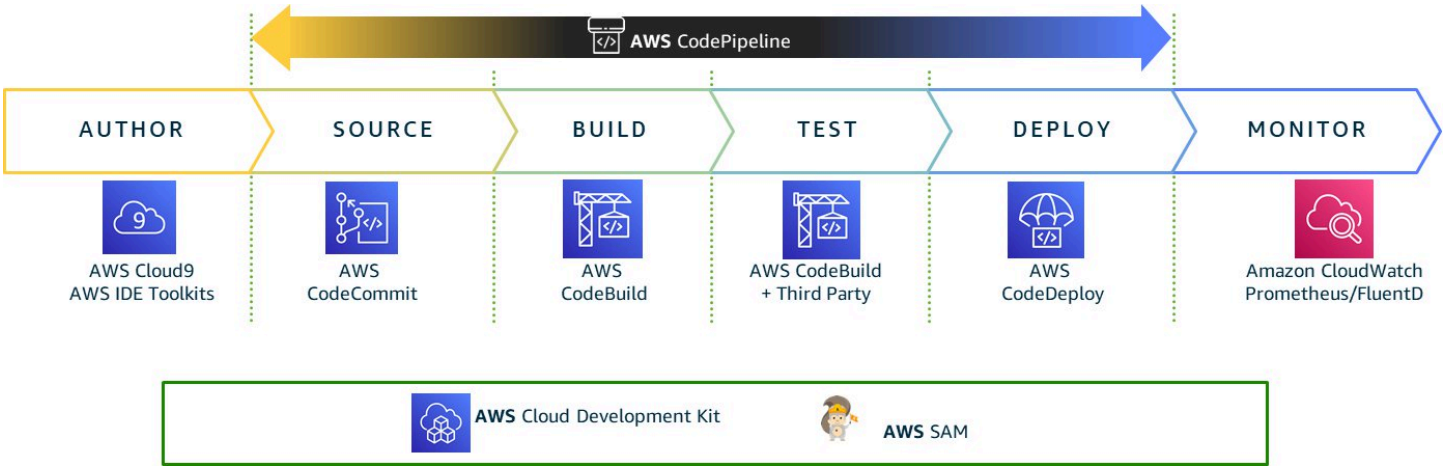
CI/CD 파이프라인 개요

CI/CD 파이프라인의 각 단계는 전달 프로세스에서 논리적 단위로 구성됩니다. 각 단계는 코드의 특정 측면을 확인하는 게이트 역할을 합니다. 파이프라인을 통해 코드가 진행됨에 따라 코드의 더 많은 측면이 계속 검증되기 때문에 이후 단계로 갈수록 코드의 품질이 더 높다고 가정합니다. 초기 단계에서 문제가 발견되면 코드가 파이프라인을 통해 진행되지 않습니다. 테스트 결과는 즉시 팀으로 전송되며 소프트웨어가 단계를 통과하지 못하면 이후의 모든 구축 및 릴리스가 중지됩니다.

AWS는 완전한 CI/CD 개발자 도구 세트를 제공하여 소프트웨어 개발 및 릴리스 주기를 가속화합니다. AWS CodePipeline은 정의된 릴리스 모델을 기반으로 코드가 변경될 때마다 릴리스 프로세스의 구축, 테스트 및 배포 단계를 자동화합니다. 따라서 기능 및 업데이트를 빠르고 안정적으로 제공할 수 있습니다.

코드 파이프라인을 다른 서비스와 통합할 수 있습니다. Amazon Simple Storage Service(Amazon S3)와 같은 AWS 서비스나 GitHub와 같은 서드 파티 제품을 예로 들 수 있습니다. AWS CodePipeline은 다음과 같은 다양한 개발 및 운영 사용 사례를 처리할 수 있습니다.

- AWS CodeBuild를 사용하여 코드 컴파일, 구축 및 테스트
- 컨테이너 기반 애플리케이션을 클라우드에 지속적 전달
- 네트워크 서비스 또는 특정 클라우드 네이티브 네트워크 기능에 필요한 아티팩트(예: 설명자 및 컨테이너 이미지)의 배포 전 검증
- 기준 및 회귀 테스트를 포함한 컨테이너식 네트워크 기능/가상 네트워크 기능(CNF/VNF)에 대한 기능/통합/성능 테스트
- 안정성 및 재해 복구(DR) 테스트



AWS CI/CD 파이프라인 구성 요소

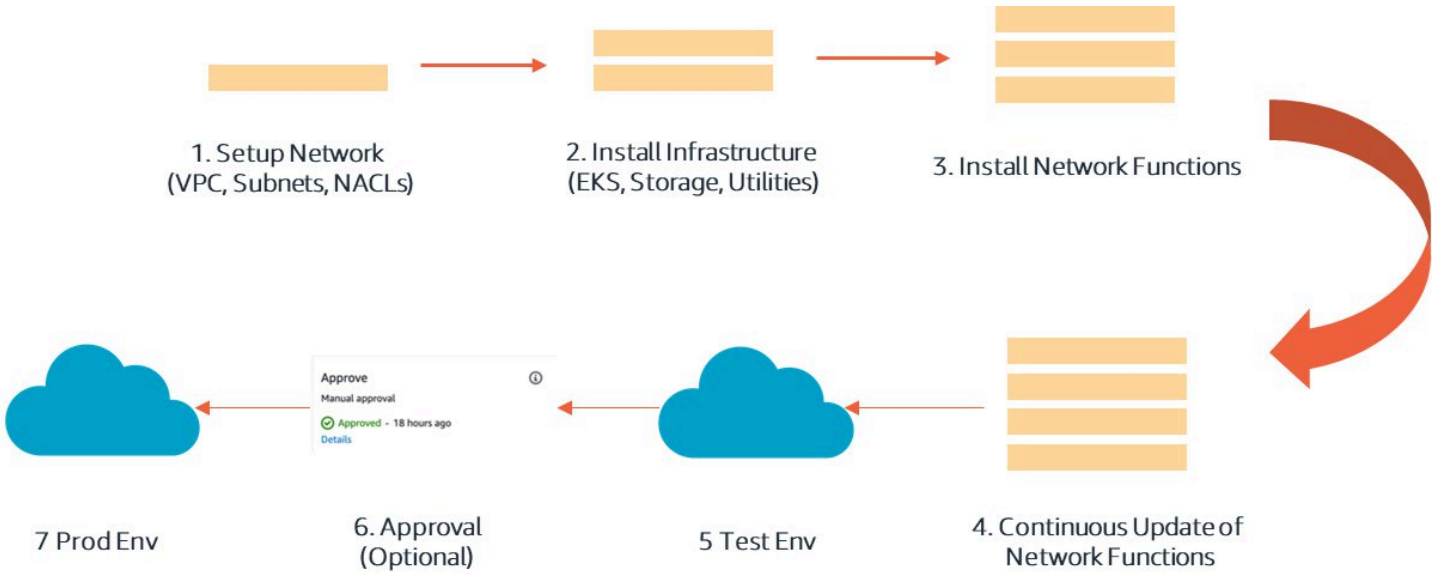
AWS는 다음 AWS 개발자 도구를 사용하여 CI/CD 파이프라인을 설정할 수 있습니다.

- AWS CodeCommit
- AWS CodeBuild
- AWS CodePipeline
- AWS CodeDeploy
- Amazon Elastic Container Registry
- AWS CodeStar

AWS CDK 및 AWS CloudFormation을 사용하여 CI/CD 파이프라인 생성을 자동화할 수 있습니다. NFV 도메인에서 이 AWS 네이티브 자동화를 관리 및 오케스트레이션(MANO) 프레임워크와 CSP의 서비스 오케스트레이션 프레임워크에 통합할 수 있습니다.

CI/CD 프로세스는 다음 단계로 이루어집니다.

- 네트워크 설정 - AWS CDK와 AWS CloudFormation이 네트워크 필수 구성 요소의 생성을 시작합니다.
    - 네트워킹 스택(VPC, 서브넷, 네트워크 주소 변환(NAT) 게이트웨이, 라우팅 테이블, 인터넷 게이트웨이)
- 인프라 배포 - AWS CDK와 AWS CloudFormation이 다음 리소스 스택의 생성을 시작합니다.
    - 컴퓨팅 스택(Amazon Elastic Kubernetes Service(Amazon EKS) 클러스터 생성, EKS 작업자 노드, AWS Lambda)
    - 스토리지 스택(Amazon S3 버킷, Amazon Elastic Block Store(Amazon EBS) 볼륨 및 Amazon Elastic File System(Amazon EFS))
    - 모니터링 스택(CloudWatch , Amazon OpenSearch Service(OpenSearch Service))
    - 보안 스택(AWS Identity and Access Management(AWS IAM), Amazon Elastic Compute Cloud(Amazon EC2) 보안 그룹, VPC 네트워크 액세스 제어 목록(NACL))
- 클라우드 네트워크 기능(CNF) 배포 - 이 단계에서는 CNF가 Kubectl 및 Helm 차트 도구를 사용하여 EKS 클러스터에 배포됩니다. 또한 이 단계에서는 CNF가 효율적으로 작동하는 데 필요한 특정 애플리케이션 또는 도구(예: Prometheus 또는 Fluentd)를 배포합니다. CNF는 Lambda 함수를 통해 배포하거나 AWS CodeBuild를 통해 배포할 수 있습니다.
- 지속적 업데이트 및 배포 - 컨테이너의 일부 변경 사항 및 구성 변경 사항을 배포하기 위해 반복적으로 수행되는 일련의 단계이며 그 결과로 업그레이드가 수행됩니다. CNF 배포 사례와 마찬가지로 AWS CodeCommit, Amazon Elastic Container Registry(Amazon ECR) 또는 GitLab 웹후크와 같은 서드 파티 소스 시스템의 트리거와 함께 AWS 서비스를 사용하여 지속적 업데이트 및 배포를 자동화할 수 있습니다.



AWS CI/CD 파이프라인 흐름도

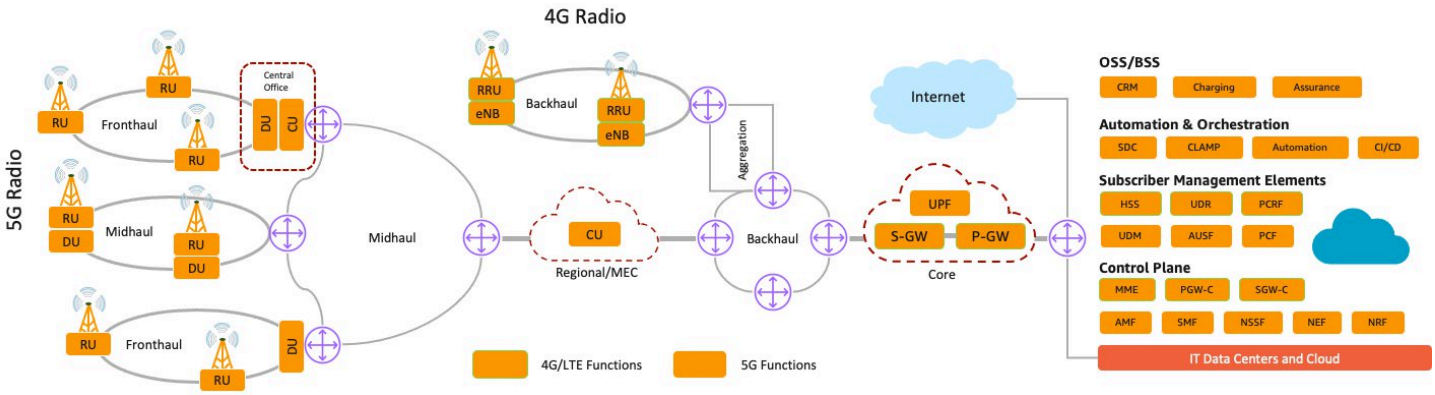
CI/CD 파이프라인은 AWS CodePipeline을 사용하여 구축되며 소프트웨어 릴리스에 필요한 단계를 모델링, 시각화 및 자동화하는 지속적 전달 서비스를 활용합니다. 파이프라인에 단계를 정의하면 소스 코드 리포지토리에서 코드를 검색하고, 해당 소스 코드를 릴리스 가능한 아티팩트로 구축하고, 아티팩트를 테스트하고, 프로덕션에 배포할 수 있습니다. 이러한 모든 단계를 성공적으로 통과한 코드만 배포됩니다. 필요한 경우 수동 승인과 같은 다른 요구 사항을 파이프라인에 추가하여 승인된 변경 사항만 프로덕션에 배포할 수 있습니다.

AWS 기반 5G 네트워크

5G 네트워크 인프라의 일반적인 모델은 4G/5G 무선 사이트, 프론트홀/미드홀/백홀 네트워크, 핵심 네트워크 사이트 및 통신/IT 데이터 센터로 구성됩니다. CSP는 AWS 서비스를 사용하여 초기 투자 비용을 줄이면서 확장 가능하고 유연한 5G 네트워크 인프라를 구축할 수 있습니다. AWS를 사용하면 운영 지원 시스템/비즈니스 지원 시스템(OSS/BSS) 및 대부분의 제어 영역 핵심 네트워크 기능을 호스팅하는 리전에 가상 네트워크 운영 센터(NOC)를 구현할 수 있습니다.

사용자 영역 기능(UPF), RAN 중앙 장치(CU) 및 Multi-Access Edge Computing(MEC)과 같은 사용자 영역 기능을 주로 호스팅하는 AWS Outposts 인스턴스 플릿을 사용하여 로컬 CO(중앙 사무실) 또는 분산 데이터 센터를 구현할 수도 있습니다. 참조 아키텍처와 AWS의 5G 네트워크 구현의 이점에 대한 자세한 설명은 5G Network Evolution on AWS(AWS를 기반으로 한 5G 네트워크 진화) 백서에 설명되어 있습니다.

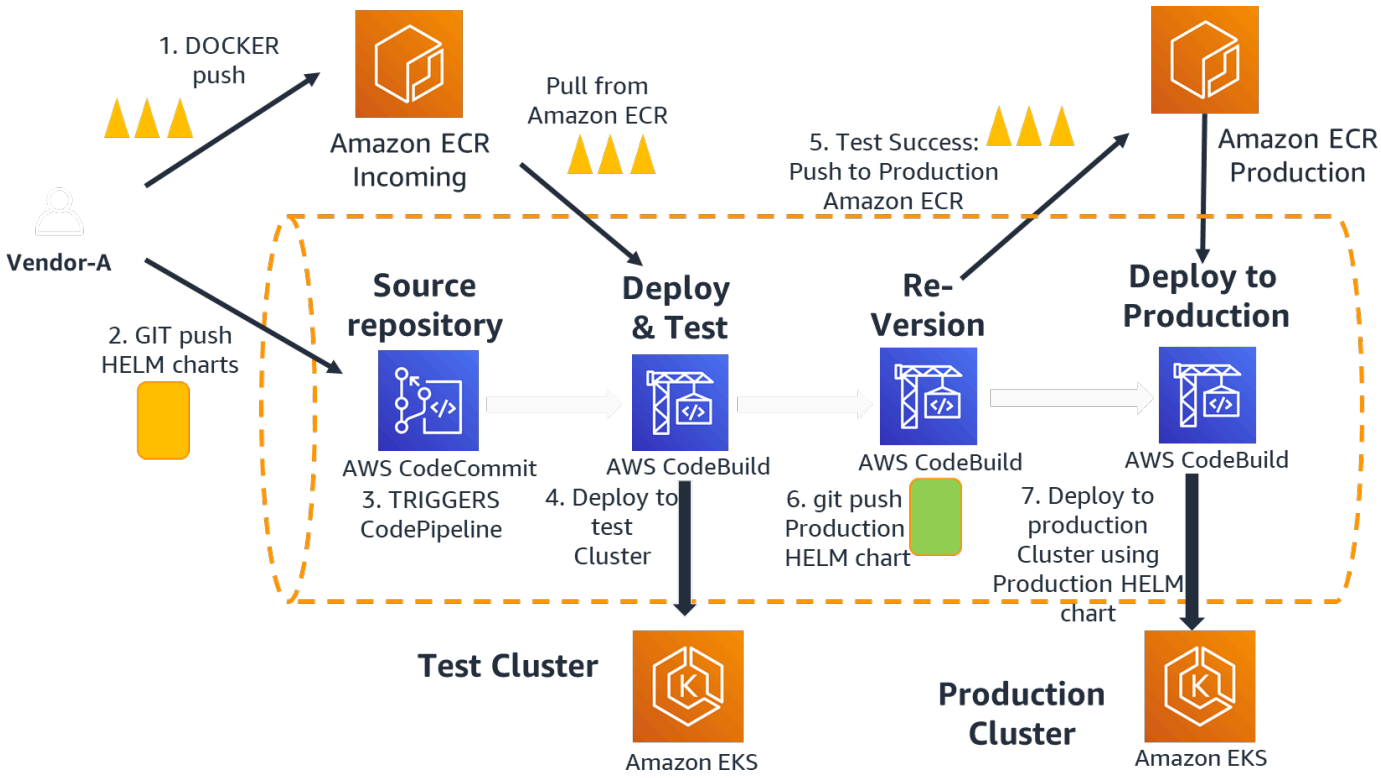
AWS 기반 5G 네트워크를 구현할 때가 되면 이 백서의 다음 단원에 소개된 AWS CI/CD 도구를 통해 5G 네트워크 기능의 배포, 업그레이드 및 수명 주기 관리를 완벽하게 자동화할 수 있습니다.



5G 네트워크 E2E 아키텍처

5G 네트워크의 CI/CD

인프라의 설계 구조는 선언적 언어를 사용하여 코드 형식으로 저장됩니다. 이를 통해 CSP는 필요에 따라 동일한 예상 동작을 사용하여 인프라를 반복적으로 재현할 수 있습니다. 코드는 코드 리포지토리에서 유지 관리되고 파이프라인은 배포된 스택에 대한 업데이트를 오케스트레이션하도록 설정됩니다(예:AWS CDK 및 AWS CloudFormation). AWS는 Independent Software Vendor(ISV) 기능의 민첩한 온보딩을 위해 코드형 인프라(IaC)를 구축할 수 있도록 지원합니다.



코드 파이프라인 흐름

Helm 차트를 통한 클라우드 네이티브 네트워크 기능 구성의 변경은 네트워크 기능에 대한 자동 CI/CD 파이프라인 실행을 위한 트리거로 간주됩니다.

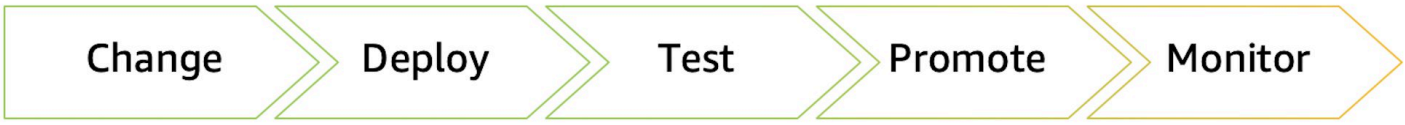
AWS CodeCommit을 사용하여 구성 파일을 유지 관리할 수 있으며, Amazon ECR을 사용하여 컨테이너 이미지를 보관할 수 있습니다.

코드 파이프라인 흐름 그림에서 볼 수 있듯이 ISV가 새로운 코드 변경 사항을 코드 리포지토리(Helm 차트, 구성 파일 또는 속성 파일)에 푸시하면 코드 파이프라인이 트리거됩니다. 코드 파이프라인은 ECR에서 이미지를 가져오고 Helm 차트를 사용하여 애플리케이션을 배포합니다. 새로운 애플리케이션 테스트를 서드 파티 테스트 자동화 프레임워크와 통합할 수 있습니다. 결과에 따라 CSP가 프로덕션 배포를 승인할 수 있습니다.

코드 파이프라인의 소스 스테이지는 구성 파일의 변경 사항을 찾습니다. 소스 스테이지에 유효한 공급자는 CodeCommit, Amazon S3, GitHub 또는 AWS CloudFormation입니다. Lambda 함수를 사용하여 웹후크를 구현함으로써 대체 소스 시스템을 통합할 수 있으며, 이를 통해 GitLab과 AWS CodePipeline 간의 이벤트 기반 통합이 가능합니다. 자세한 구현 안내서는 다음 링크를 참조하세요.

- GitLab을 사용한 웹후크
- 컨테이너 레지스트리 통합

CI/CD 파이프라인을 설계할 때는 테스트 결과가 예상과 일치하고 기준에 부합하는지 검증한 후 초기 배포, 테스트 및 프로덕션으로의 승격과 같은 중요한 배포 단계를 고려해야 합니다. 파이프라인 프로세스의 모든 단계에서 데이터 아티팩트가 제공되므로 비교 및 데이터 중심 의사 결정이 가능합니다.



애플리케이션 CI/CD 파이프라인 단계

모든 단계는 별도의 작업으로 간주되므로 네트워크 서비스 및 클라우드 네이티브 네트워크 기능을 지원하는 데 적합한 검증 및 배포 워크플로를 통합할 수 있습니다. 실행 작업은 트래픽 생성기 및 시뮬레이터와 같은 추가 서드 파티 도구를 통합하여 엔드 투 엔드 네트워크 서비스 검증을 가능하게 할 수 있습니다.

AWS는 다른 AWS 서비스와 기본적으로 통합되는 정교한 AWS Step Functions(클라우드 네이티브 상태 시스템) 서비스를 제공하며 Jira 또는 테스트 자동화 프레임워크와 같은 외부 시스템과도 통합할 수 있습니다.

CI/CD 세부 단계

CI/CD는 새 코드가 한쪽 끝에서 제출되고 일련의 단계(소스, 구축, 테스트, 스테이징 및 프로덕션)에 걸쳐 테스트된 후 프로덕션에서 즉시 사용 가능한 코드로 게시되는 파이프라인으로 나타낼 수 있습니다.

배포 및 테스트 단계는 다음과 같습니다. 배포 및 구성은 다음 네 가지 주요 섹션으로 나뉩니다.

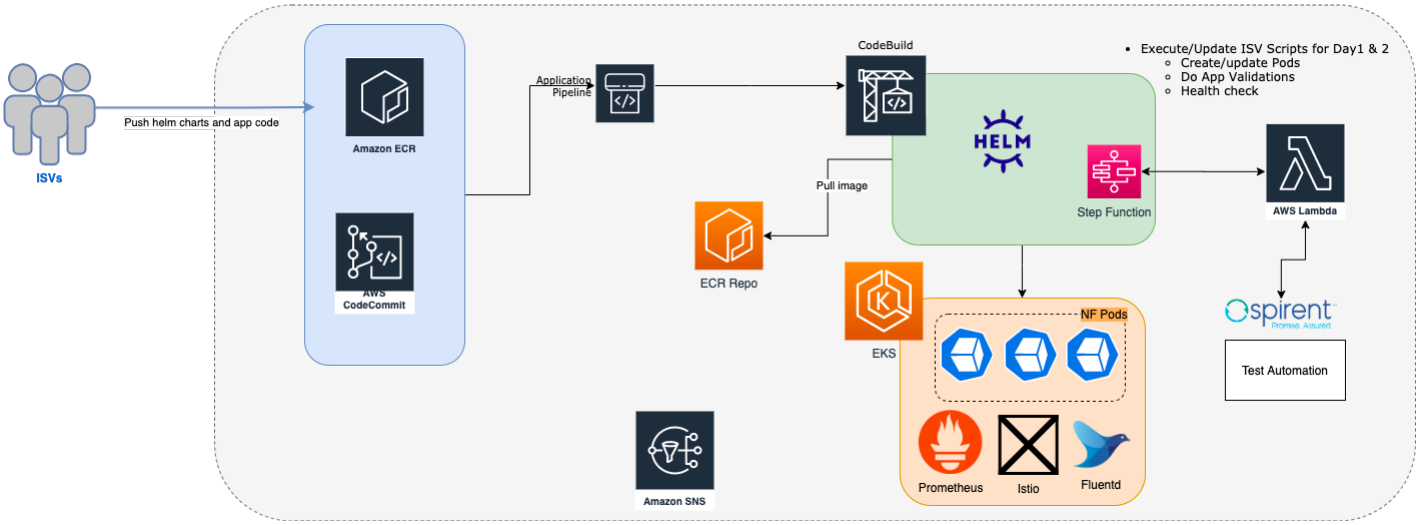
- 네트워크 설정
- 인프라 배포
- 클라우드 네이티브 네트워크 기능 배포
- CFN 지속적 전달

네트워크 설정

인프라 필수 구성 요소 설정에 중점을 둡니다. 여기에는 VPC, 네트워크, 서브넷, NACL 등의 생성이 포함됩니다. ISV 및 고객의 구현 계획(예: 다중 테넌시 및 정적 할당 대 동적 할당)을 고려하여 IP 네트워크 계획을 설계합니다. 이 계획은 AWS CDK 또는 AWS CloudFormation으로 코딩할 수 있습니다. 이 코드를 실행하면 클라우드 인프라 네트워크 필수 구성 요소가 배포됩니다.

인프라 배포

인프라 배포는 인프라 구성 요소를 프로비저닝합니다. 여기에는 EKS 클러스터 및 지원 인프라(예: EFS, EKS 작업자 노드, ELB)를 생성하고 클라우드 네이티브 네트워크 기능 요구 사항에 따라 클러스터를 구성하는 작업이 포함됩니다. CNF 요구 사항에 따라 AWS는 Multus 인터페이스를 포함하여 노드에 대한 추가 네트워크 인터페이스도 배포합니다. 대부분의 배포 및 구성 단계는 애플리케이션에 대한 일회성 작업이며 필요한 경우에만 애플리케이션에 대한 업데이트로 업데이트됩니다.

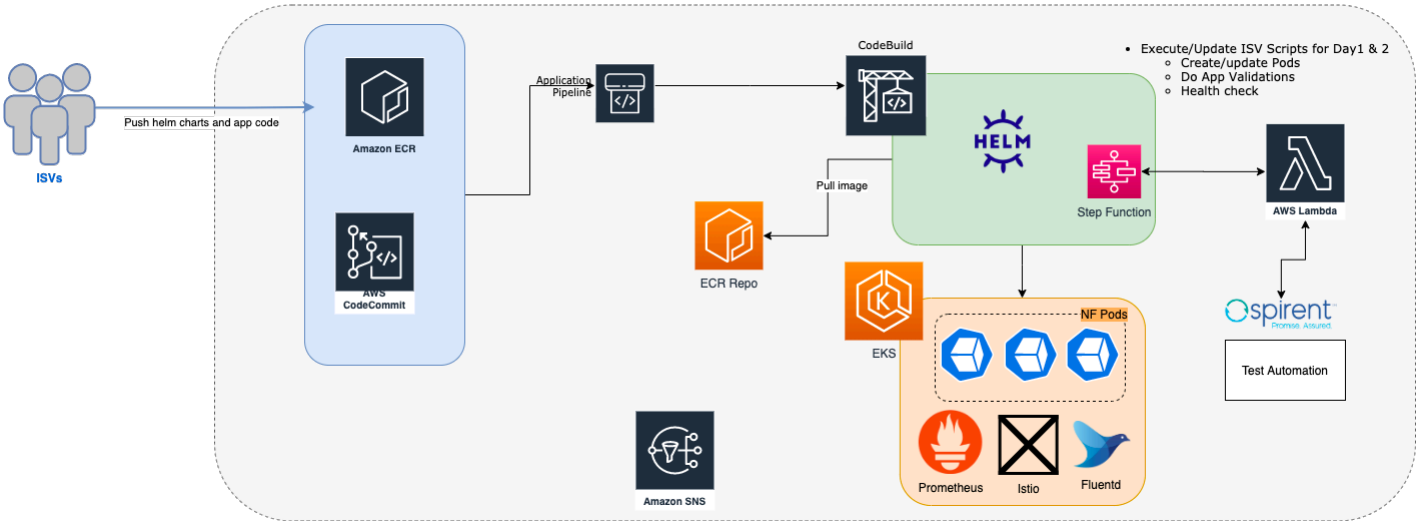


CDK를 사용한 인프라 배포

클라우드 네이티브 네트워크 기능 배포

CNF 배포는 애플리케이션 배포와 관련됩니다. CNF 배포의 일환으로 애플리케이션의 Helm 차트가 CI/CD 코드 파이프라인을 통해 구현됩니다. 주로 사전 및 사후 검사를 포함하는 개별 애플리케이션별 스크립트를 실행하기 위한 콜백이 통합됩니다. Helm 차트는 애플리케이션의 필요에 따라 순서대로 구현되며, 배포의 다음 단계로 이동하기 전에 Kubernetes 포드의 상태를 확인합니다. ISV는 Helm 차트와 안전성 검사를 실행하기 위한 래퍼 스크립트를 제공합니다. 이러한 ISV 스크립트는 AWS CodePipeline 내에서 호출됩니다. 이 단계에서는 애플리케이션의 클라우드 인프라를 로깅하고 모니터링하는 Amazon CloudWatch와 함께 Prometheus, Fluentd와 같은 로깅 및 모니터링 에이전트가 배포됩니다.

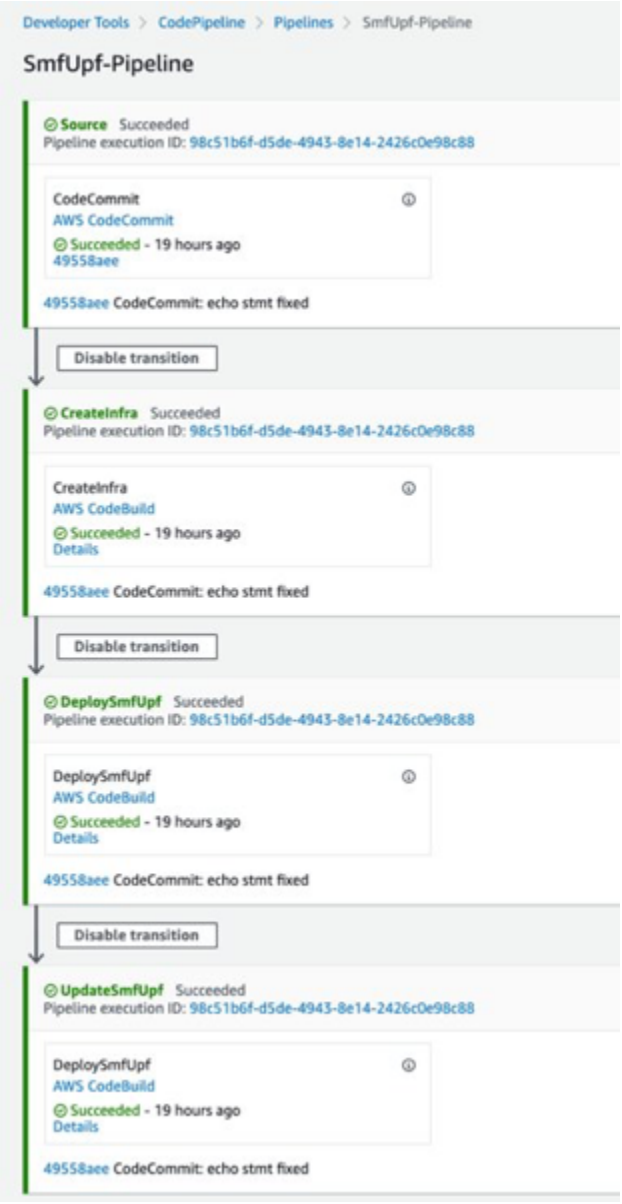
코드 파이프라인은 서드 파티 테스트 자동화 프레임워크와 통합됩니다. 코드 파이프라인은 테스트 자동화 프레임워크 API를 직접 호출하여 배포된 애플리케이션에서 테스트를 실행하고, 테스트 결과를 쿼리하며, 결과를 분석할 수 있습니다. 이를 통해 애플리케이션의 배포와 테스트가 간소화됩니다.



애플리케이션 배포 및 업데이트

다음은 AWS CodePipeline을 통해 사용자 영역 기능/세션 관리 기능(UPF/SMF) CNF를 배포하는 예입니다.

- CodeCommit, CodeBuild 및 CodePipeline을 사용하여 전체 CI/CD 프로세스를 자동화합니다.
- 인프라 생성 및 애플리케이션 설치 작업은 파이프라인의 일부로 통합됩니다.
- Fluentd 및 Prometheus 에이전트는 Amazon CloudWatch 대시보드에 설치 및 생성됩니다.



UPF/SMF CNF 배포 예

CFN 지속적 전달

이 단계는 컨테이너의 일부 변경 사항 및 구성 변경 사항을 배포하기 위해 반복적으로 수행되는 일련의 단계이며 그 결과로 업그레이드가 수행됩니다. CNF 지속적 전달은 파이프라인을 통해 자동화되며 개별 애플리케이션에만 적용됩니다. AWS는 표준 Helm 차트를 사용하여 특정 CNF를 업데이트합니다. 코드 파이프라인에는 애플리케이션 업데이트 상태에 대한 사전 및 사후 검사가 있습니다. 업데이트된 CI/CD 파이프라인은 테스트 자동화 프레임워크와 통합되어 자동화된 테스트를 실행합니다. 이러한 추상화를 통해 네트워크 기능을 깔끔하게 배포할 수 있습니다.

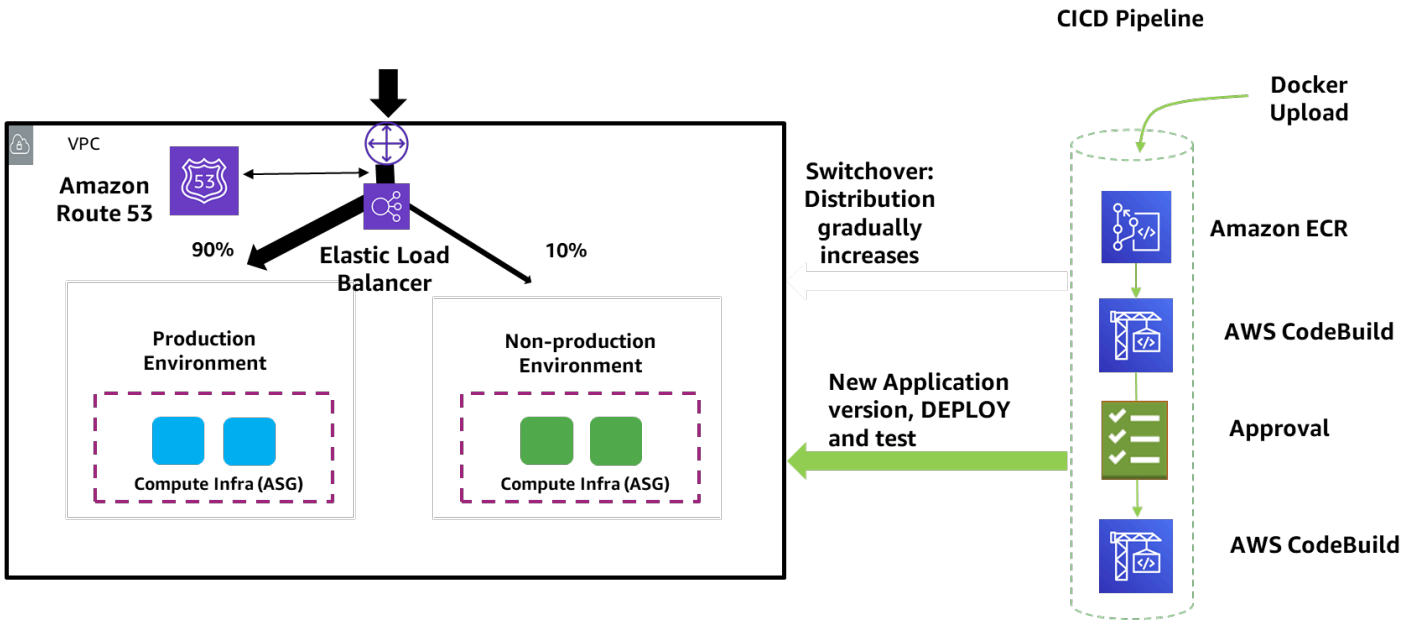
CNF 지속적 전달 및 배포는 크게 다음 범주로 분류될 수 있습니다.

- 애플리케이션 업그레이드 - 대부분의 애플리케이션 업그레이드는 Kuberbetes 애플리케이션 포드 내의 변경 사항입니다. 이러한 업데이트는 코드 파이프라인을 통해 자동으로 적용될 수 있습니다. 대부분의 CNF는 애플리케이션 포드 인스턴스를 여러 개 제공하여 현재 위치 업그레이드를 지원합니다. 여러 인스턴스를 업그레이드할 경우 롤링 업그레이드 방식을 사용할 수 있습니다. 모든 애플리케이션 포드 변경이 Helm 업그레이드를 지원하는 것은 아닙니다. 파이프라인은 이러한 변형을 고려하며 필요에 따라 Helm 설치/삭제를 사용합니다.
- 주요 업그레이드 - 주요 업그레이드는 주로 데이터베이스 스키마 변경입니다. 이 변경 사항을 적용하려면 다소 간의 가동 중지 시간을 감수해야 합니다. 이러한 변경에 대한 표준 접근 방식은 애플리케이션을 삭제하고 관련 포드를 다시 생성하는 것입니다. 이 프로세스 동안 애플리케이션을 사용하지 못할 수도 있습니다. 업그레이드에 사용되는 도구는 다음과 같습니다.

    - AWS CloudFormation - 고객이 JSON 또는 YAML 템플릿으로 모든 인프라 리소스를 설명하고 프로비저닝할 수 있습니다. AWS CloudFormation은 Lambda 지원 사용자 정의 리소스를 통해 강력한 확장 메커니즘을 제공합니다. 고객이 AWS CloudFormation을 AWS 리소스 이상으로 확장하고 필요한 리소스를 다른 환경에 프로비저닝할 수 있습니다(예: 하이브리드 환경에 온프레미스 리소스 프로비저닝). AWS CDK는 개발자에게 Python, TypeScript, JavaScript, Java 및 C#과 같은 높은 수준의 친숙한 프로그래밍 언어를 사용하여 코드를 구축한 후 하위 수준 AWS CloudFormation JSON 형식으로 컴파일하여 배포할 수 있는 기능을 제공합니다.
    - 블루/그린 배포 - AWS는 테스트 환경과 프로덕션 환경에서 블루/그린 및 카나리 기반 배포를 지원하고 권장합니다. 블루/그린 배포를 사용하면 포함된 환경에서 새 애플리케이션 버전을 테스트할 수 있습니다. 이 배포는 운영 트래픽을 전환할 수 있는 쉽고 효율적인 방법을 제공합니다. 카나리 기반 배포는 프로덕션 트래픽으로 인해 발생하는 문제를 발견하기 위해 소량의 프로덕션 트래픽으로 비프로덕션 그린 환경을 테스트하여 이 개념을 확장합니다. 새로운 애플리케이션 버전은 내부 시뮬레이션 테스트 트래픽과 소량의 프로덕션 트래픽을 사용하여 테스트되므로 사용자는 확신을 가지고 프로덕션 트래픽으로 전환할 수 있습니다. 프로덕션 트래픽은 전환이 완료될 때까지 서서히 증가합니다. 구현에는 가중치 기반 DNS와 가중치 기반 ELB 대상 그룹이 포함됩니다.
    - 블루/그린 및 카나리 기반 배포 단계로 AWS CodePipeline을 구성하여 자동화를 달성할 수 있습니다. 승인 단계는 처음에는 프로비저닝 중에 수동으로 실행할 수 있지만 나중에는 완전히 자동화해야 합니다. 테스트 환경에서는 프로덕션에 배포하기 전에 항상 롤백 작업으로 테스트하여 이후 버전 및 이전 버전과의 호환성을 검증하는 것이 좋습니다. 서비스 메시가 있는 클러스터의 블루/그린 배포는 정상적인 전환을 수행하기 위

해 서비스 메시에 대한 최종 애플리케이션 및 라우팅 게이트웨이에서 제공하는 지원에 따라 달라집니다.

- AWS Systems Manager는 통합 사용자 인터페이스를 제공하므로 CI/CD로 배포된 네트워크 기능에 사용되는 여러 AWS 서비스의 운영 데이터를 볼 수 있습니다. Systems Manager를 사용하면 AWS 리소스 전체에서 운영 작업을 자동화할 수 있습니다.



카나리 배포

보안

보안은 중요한 요소입니다. 다음은 애플리케이션을 배포할 때 AWS CI/CD 프로세스에서 고려할 보안 단계의 목록입니다.

- 소스 - 공급 업체에 할당된 ECR 리포지토리는 '푸시할 때 스캔(Scan on Push)' 플래그가 활성화되어 있으므로 Docker 이미지 업로드 시 즉시 보안 검사를 받게 됩니다. 알려진 모든 CVE(일반적인 취약성 및 노출도)도 알림으로 플래그가 지정됩니다. ECR 외에도 공급 업체는 차트를 AWS CodeCommit 리포지토리에 저장할 때 일반 텍스트를 사용하는 대신 Secrets Manager를 통해 사용된 모든 암호를 암호화하라는 요청을 받습니다.
- 아티팩트 무결성 - 파이프라인에서 사용되는 아티팩트는 저장되었는지(AWS 관리형 키 사용) 또는 전송 중인지(SSL/TLS 사용) 여부에 관계없이 암호화됩니다.
- IAM 사용자 및 역할 - 사용자 또는 리소스에 제공되는 권한은 최소 권한의 원칙을 기반으로 합니다. 서로 다른 서비스의 리소스로 작업하는 경우 구성해야 하는 교차 IAM 역할 신뢰 관계가 있어야 합니

다. 예를 들어, AWS CodeBuild에는 Amazon EKS 클러스터에서 명령을 실행할 수 있는 권한이 필요합니다.

- 감사 - AWS CloudTrail에서 제공하는 감사 기능은 서비스 및 사용자 작업 전반의 모든 API 호출을 추적하므로 과거 이벤트를 평가할 수 있습니다.
- 이미지 취약성 검사 - Amazon ECR에 업로드된 CNF 이미지는 자동으로 보안 취약성이 검사됩니다. 검사 결과에 대한 보고서는 AWS Management Console에서 확인할 수 있으며 API를 통해 검색할 수도 있습니다. 그런 다음 CNF 이미지 교체를 포함한 시정 조치를 위해 결과를 CSP 운영자에게 보낼 수 있습니다.

보안 검사는 파이프라인의 여러 단계에서 수행되어 새로 업로드된 이미지가 안전하며 원하는 규정 준수 검사를 준수하는지 확인하고 승인을 위해 CSP에 알림을 보낼 수 있습니다.

- 컨테이너 레지스트리는 알려진 CVE 취약성을 검사합니다.
- 구성은 테스트 단계 동안 알려진 개인 식별 정보(PII) 패턴인 정보 유출을 검사하여 예기치 않은 개방형 TCP/UDP 포트 및 DOS 취약성과 같은 문제에 대한 규정 준수 검사 규칙을 트리거합니다.
- 안전한 업그레이드/롤백을 위해 이전 버전 및 이후 버전과의 호환성이 검증됩니다.

애플리케이션 외에도 저장되었는지 전송 중인지 여부에 관계없이 전 단계에서 아티팩트의 암호화된 전송을 보장하여 파이프라인 보안을 프로비저닝하는 것이 중요합니다.

관찰

AWS는 기본적으로 AWS에 배포된 5G CNF에 대한 관찰 기능을 활성화합니다. 이 기능은 Amazon CloudWatch에 의해 활성화됩니다. CloudWatch는 클라우드 리소스 및 애플리케이션에 대한 완벽한 가시성을 제공합니다.

Amazon CloudWatch는 이 프로세스 동안 네 가지 주요 단계를 수행합니다.

1. 수집 - AWS 및 온프레미스 서버에서 실행되는 모든 AWS 리소스, 애플리케이션 및 서비스에서 지표와 로그를 수집합니다.
2. 모니터링 - CloudWatch 대시보드를 사용하여 애플리케이션과 인프라를 시각화하고, 로그와 지표를 나란히 연결하여 문제를 해결하고, CloudWatch 경보를 사용하여 알림을 설정합니다.
3. 조치 - CloudWatch Events 및 AWS Auto Scaling을 사용하여 운영 변경 사항에 대한 대응을 자동화합니다.
4. 분석 - CloudWatch Metric Math를 사용하여 최대 1초의 지표, 연장된 데이터 보존(15분) 및 실시간 분석을 제공합니다.

Amazon CloudWatch 에이전트는 고객의 Kubernetes 클러스터에 설치됩니다. 이 에이전트는 Prometheus 구성, 검색 및 지표 가져오기 기능을 지원하여 충실도가 높은 Prometheus 지표 및 메타데이터를 모두 내장 지표 형식(EMF)으로 보강하고 CloudWatch Logs에 게시합니다.

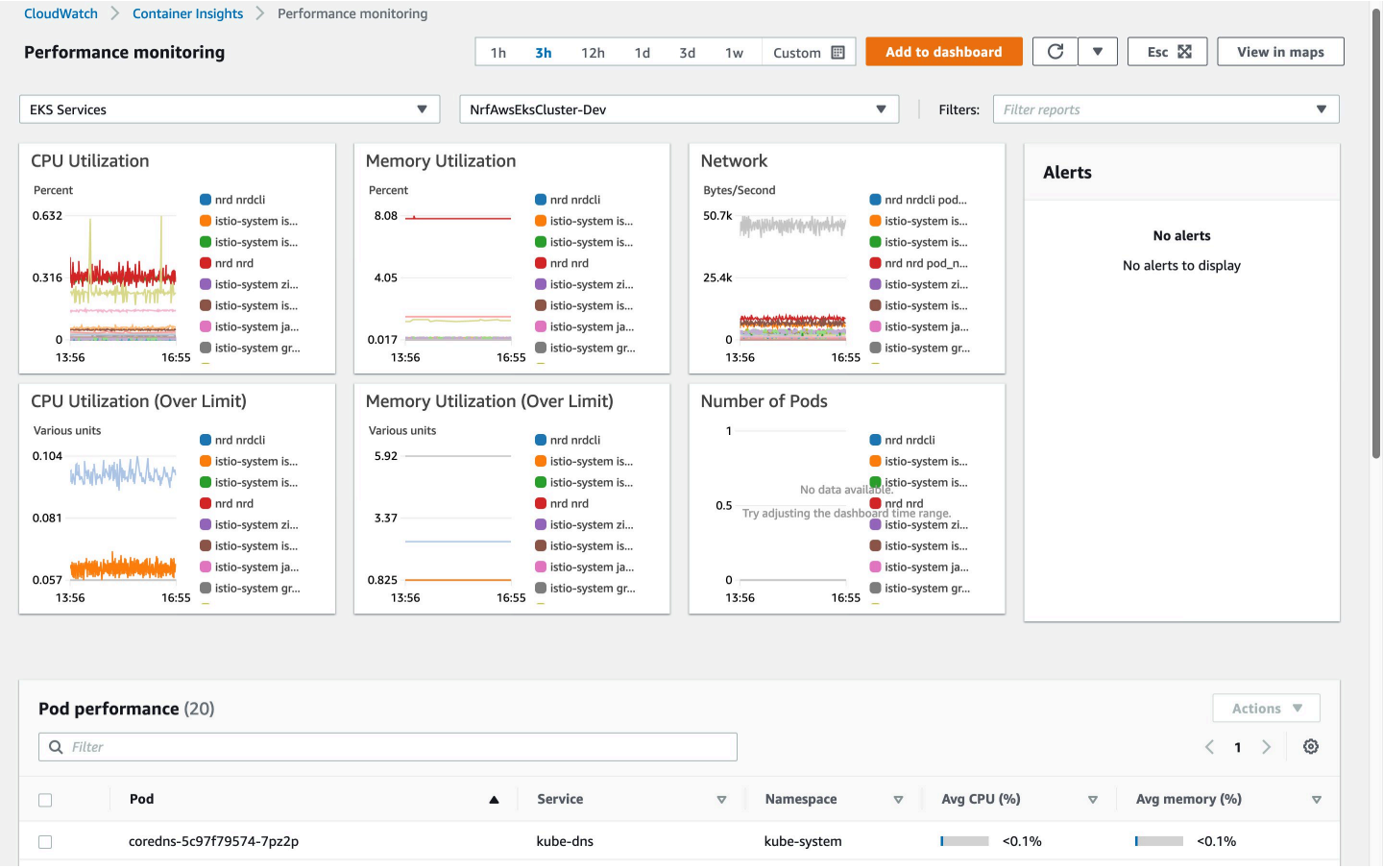
Amazon CloudWatch Container Insights는 컨테이너식 애플리케이션에서 Prometheus 지표를 검색하고 수집하는 작업을 자동화합니다. 대시보드에 시각화된 집계된 사용자 정의 CloudWatch 지표를 자동으로 수집, 필터링 및 생성합니다.

각 이벤트는 완전히 구성 가능한 큐레이팅된 지표 차원 세트에 대한 CloudWatch 사용자 정의 지표로 지표 데이터 요소를 생성합니다. 집계된 Prometheus 지표를 CloudWatch 사용자 정의 지표 통계로 게시하면 성능 문제와 장애를 모니터링하고 경보를 보내고 해결하는 데 필요한 지표 수가 줄어듭니다. 또한 CloudWatch Logs Insights 쿼리 언어로 충실도가 높은 Prometheus 지표를 분석하여 컨테이너식 환경의 상태 및 성능에 영향을 미치는 특정 포드 및 레이블을 격리할 수 있습니다.

AWS CloudTrail은 서비스 전반에 걸친 모든 API 호출을 기록하여 이러한 가시성을 제공합니다. AWS Config는 규정 준수를 검증하는 기능을 제공합니다. AWS는 AWS X-Ray 및 AWS CloudTrail과 같은 다양한 서비스를 사용하여 지표, 로그 및 이벤트(애플리케이션, 인프라 및 파이프라인 관련 이벤트)에 대한 추가 모니터링 옵션을 제공합니다.

- AWS는 기본적으로 Prometheus, Fluentd 등과 같은 오픈 소스 지표 도구를 통합할 수 있습니다.
- Prometheus 지표는 Amazon CloudWatch에 추가로 수집하거나 추가 분석을 위해 OpenSearch Service에 수집할 수 있습니다.
- AWS는 Fluentd를 표준 메커니즘으로 사용하여 다양한 시스템에서 로그를 수집합니다. 이 프로젝트에도 동일한 메커니즘이 사용되고 구성됩니다.

이 메커니즘을 구성하는 방법에 대한 자세한 내용은 Fluentd를 DaemonSet로 설정하여 CloudWatch Logs에 로그 전송을 참조하세요.



Amazon CloudWatch에서 모니터링하는 지표의 예

서드 파티 및 오픈 소스 도구를 사용한 CI/CD 오케스트레이션

오케스트레이션 계층은 IaC를 사용하여 5G 네트워크 기능을 실행하는 데 필요한 기본 인프라를 배포하고 구성합니다. 이 계층은 모듈식이고, 휴대 및 재사용이 가능하도록 설계되어야 합니다.

인프라는 클라우드 네이티브 모범 사례를 따르며 가용성, 중복성 및 확장성이 뛰어납니다.

이전 단원에서 설명한 것처럼 기본 인프라의 배포는 AWS Cloud Development Kit를 사용하여 수행할 수 있습니다. 또한 Hashicorp의 Terraform을 사용하여 수행할 수도 있습니다.

Terraform

Terraform은 수백 개의 클라우드 서비스를 관리하기 위한 일관된 Command Line Interface(CLI) 워크플로를 제공하는 오픈 소스 IaC 소프트웨어 도구로, 클라우드 API를 선언적 구성 파일로 코드화합니다.

Terraform을 사용하여 배포하는 경우 CDK에 사용된 것과 동일한 원칙을 사용합니다. 코드는 공급 업체 요구 사항에 따라 네트워킹 구성 요소를 사용자 정의하고 재사용할 수 있는 모듈로 구성됩니다.

구성은 모두 파라미터화되어 있으므로 제공업체 및 ISV 권장 사항에 따라 배포를 완벽하게 조정할 수 있습니다.

네트워크 기능 배포는 두 단계로 구분됩니다.

- 필요한 AWS 인프라는 중앙 리포지토리를 통해 생성 및 관리됩니다.
- 구성 및 코드는 GitHub 리포지토리에 중앙 집중식으로 저장됩니다.

필수 구성 요소가 생성되면 이전 단계에서 설정한 애플리케이션 파이프라인을 사용하여 네트워크 기능을 배포할 준비가 됩니다.

인프라 배포

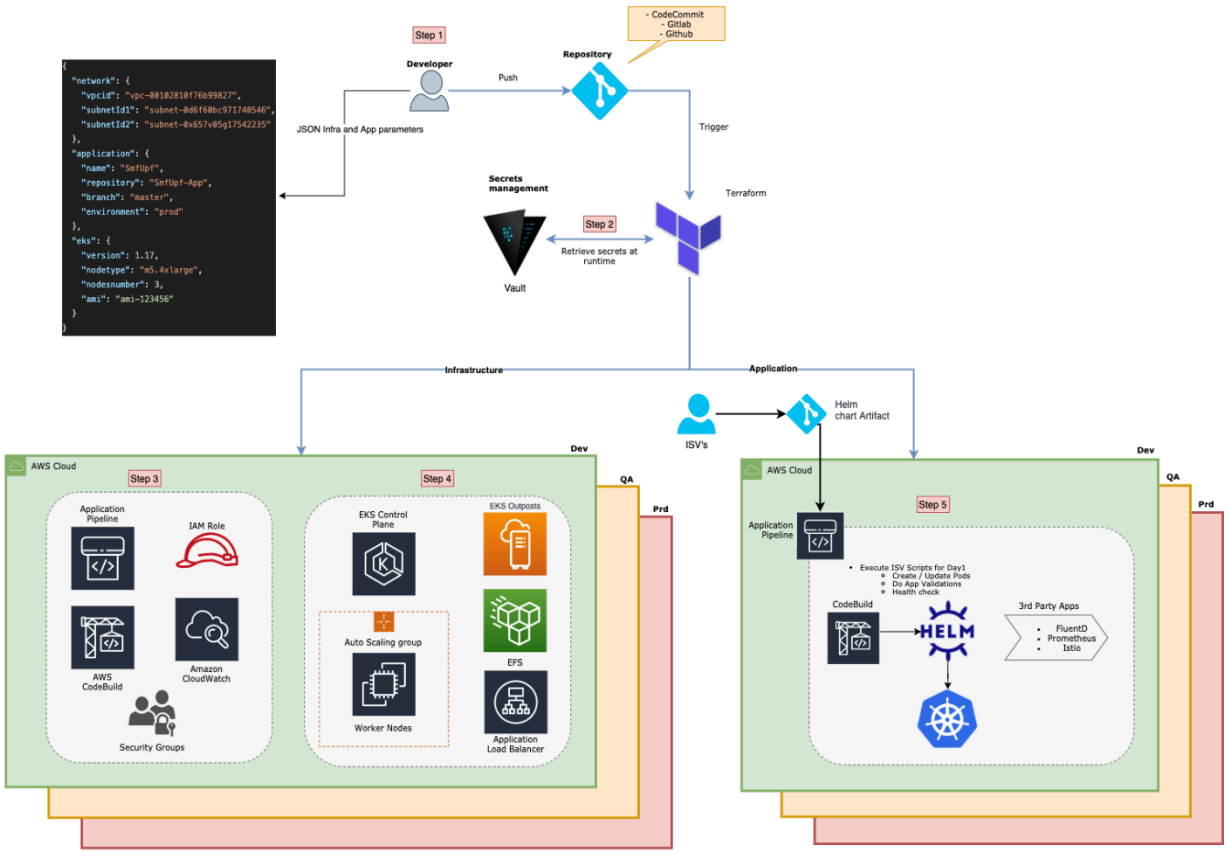
인프라 배포에는 네트워크 기능을 성공적으로 배포 및 구성하기 위한 모든 필수 구성 요소가 포함됩니다.

이 단계에서 생성되는 몇몇 구성 요소는 다음과 같습니다.

- 네트워킹 - VPC, 퍼블릭 및 프라이빗 서브넷, 경로, 로드 밸런서
- 컴퓨팅 - Kubernetes(Vmware Tanzu, Amazon EKS 또는 AWS Outposts), Amazon EC2 인스턴스 주 노드 및 작업자 노드, Auto Scaling 그룹
- 스토리지 - Amazon EFS, Amazon EBS, Amazon S3 버킷
- 보안 - IAM 역할, 보안 그룹
- 파이프라인 - CodePipeline, CodeBuild
- 관찰 - CloudWatch, Prometheus, Fluentd

다음은 Terraform에서 오케스트레이션한 인프라 시퀀스입니다. 아래 그림을 참조하세요.

1. 개발자가 중앙 리포지토리에 저장된 JSON 파일을 IaC 코드로 채웁니다. 이 파일에는 인스턴스 크기, Kubernetes 버전, 네트워크 정보 및 애플리케이션 리포지토리 세부 정보와 같은 원하는 인프라 구성에 대한 정보가 포함되어 있습니다.
2. 런타임에 HashiCorp Vault 또는 AWS Secrets Manager에서 보안 정보를 검색합니다.
3. 인프라 구성 요소(네트워킹, 컴퓨팅, 스토리지 및 보안)를 배포하고 구성합니다.
4. 네트워크 기능 포드를 호스팅하는 작업자 노드가 있는 Amazon EKS 클러스터가 배포됩니다. Amazon EKS를 AWS Outposts에 배포하여 데이터 센터에 근접해야 하는 워크로드를 지원할 수도 있습니다.
5. 네트워크 기능 리포지토리의 변경 사항을 수신하도록 애플리케이션 파이프라인이 생성되고 구성됩니다. 구성된 리포지토리 브랜치로 코드가 푸시될 때마다 파이프라인은 네트워크 기능의 구축, 테스트 및 배포를 자동으로 트리거합니다.
6. 로그와 지표를 수집하고 중앙 집중화하는 관찰 도구는 모든 노드에 서비스로 배포되며, Grafana 또는 OpenSearch Dashboards에서 시각화할 수 있는 거의 실시간 데이터를 제공합니다.



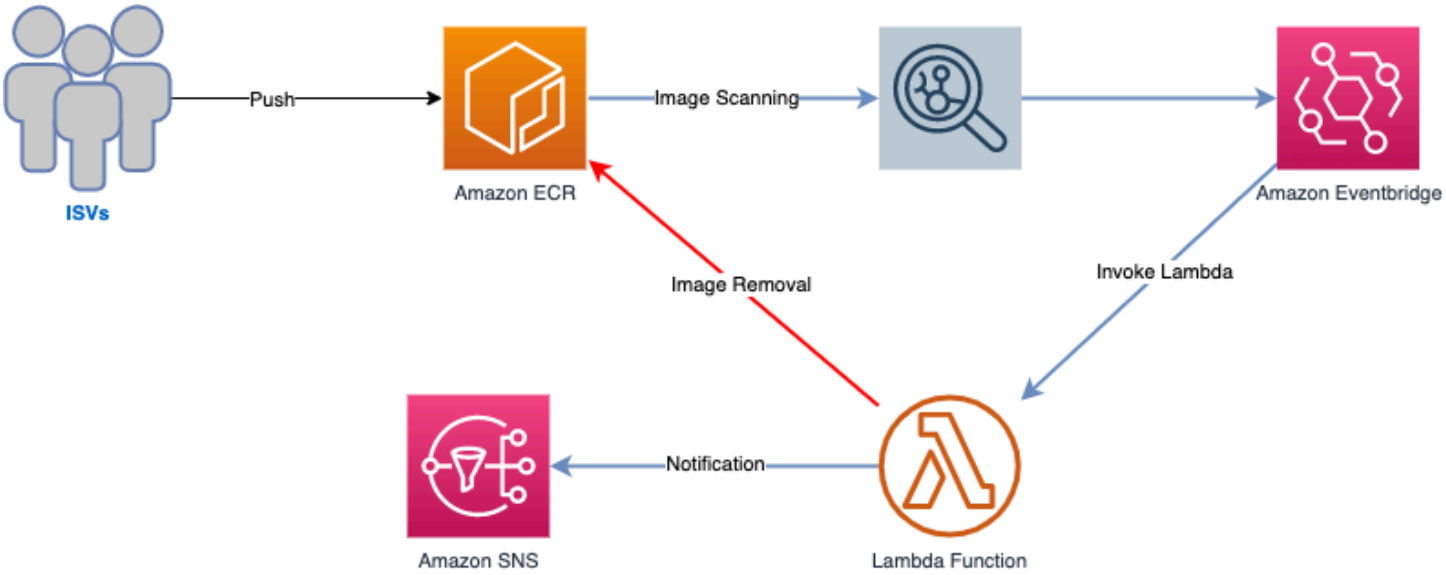
네트워크 기능 배포 및 구성

네트워크 기능 배포 및 구성

ISV와 제공업체는 이전 단계에서 생성된 파이프라인을 통해 네트워크 기능 배포를 분산하고 최적화할 수 있습니다. 파이프라인이 연결되고 애플리케이션 리포지토리의 변경 사항을 수신합니다. 이 변경 사항은 이전 그림의 1단계에서 JSON 파일에 구성되어 있습니다.

서드 파티에서 게시한 이미지를 검사할 수 있도록 컨테이너 이미지의 소프트웨어 취약성을 식별하는 데 도움이 되는 취약성 검사 솔루션을 배포하고 구성합니다. 검사 솔루션은 Amazon ECR에 푸시된 모든 새 이미지를 자동으로 검사합니다. ECR 이미지 검사에 대한 자세한 내용은 이미지 검사를 참조하세요.

다음 그림은 이미지 취약성 검사 솔루션의 아키텍처를 보여 줍니다.



이미지 취약성 검사 솔루션의 아키텍처

검사 결과 이후의 이미지 변경 또는 리포지토리의 직접 변경에 의해 트리거되도록 애플리케이션 파이프라인을 구성할 수 있습니다. 새 Helm 이미지가 생성되는 경우를 예로 들 수 있습니다.

다음 목록은 네트워크 기능을 생성/업그레이드하는 시퀀스입니다. 아래 그림을 참조하세요.

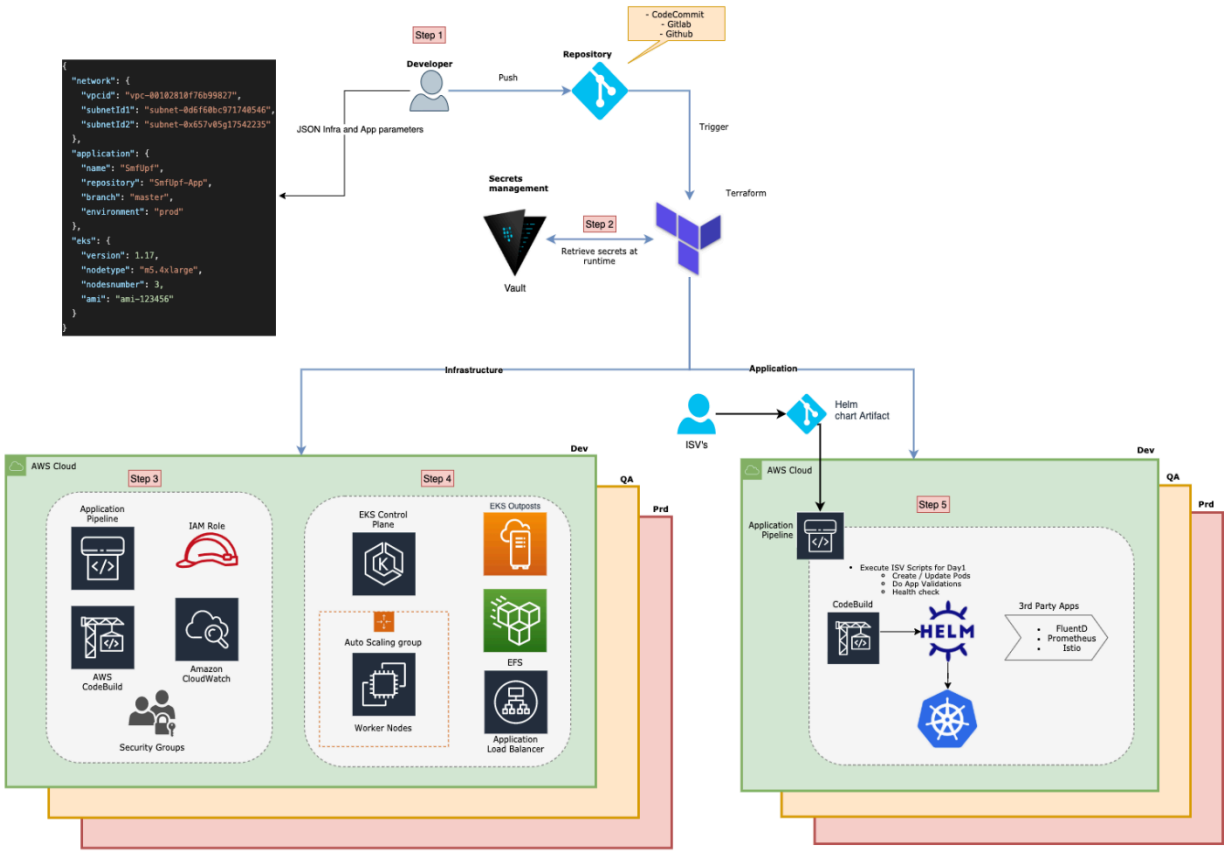
ISV가 Amazon ECR에 새 이미지를 게시합니다. 이미지가 승인되면 애플리케이션 파이프라인이 트리거됩니다.

CodePipeline이 Amazon ECR에서 새 이미지를 가져오고 CodeBuild를 사용하여 이미지를 Kubernetes에 배포합니다. Helm 명령은 네트워크 기능을 업그레이드하는 데 사용할 수 있습니다.

이미지가 배포된 후 서비스형 테스트(TaS)가 트리거됩니다. TaS는 새로운 배포를 검증하고 스트레스 상황에서 네트워크 기능의 성능에 대한 데이터 및 지표를 중앙 집중화합니다.

로그와 지표는 OpenSearch 및 Grafana에서 수집되고 중앙 집중화됩니다. Datadog, Istio 및 Prometheus와 같은 서드 파티 도구를 구성하여 추가 관찰 기능을 제공할 수도 있습니다.

네트워크 리소스를 조정할 수 있는 MANO도 배포하고 솔루션과 통합할 수 있습니다. 수집된 데이터를 사용하여 네트워크 슬라이싱 및 서비스 품질(QoS) 자동 크기 조정과 같은 자동화된 작업을 수행합니다.



애플리케이션 파이프라인

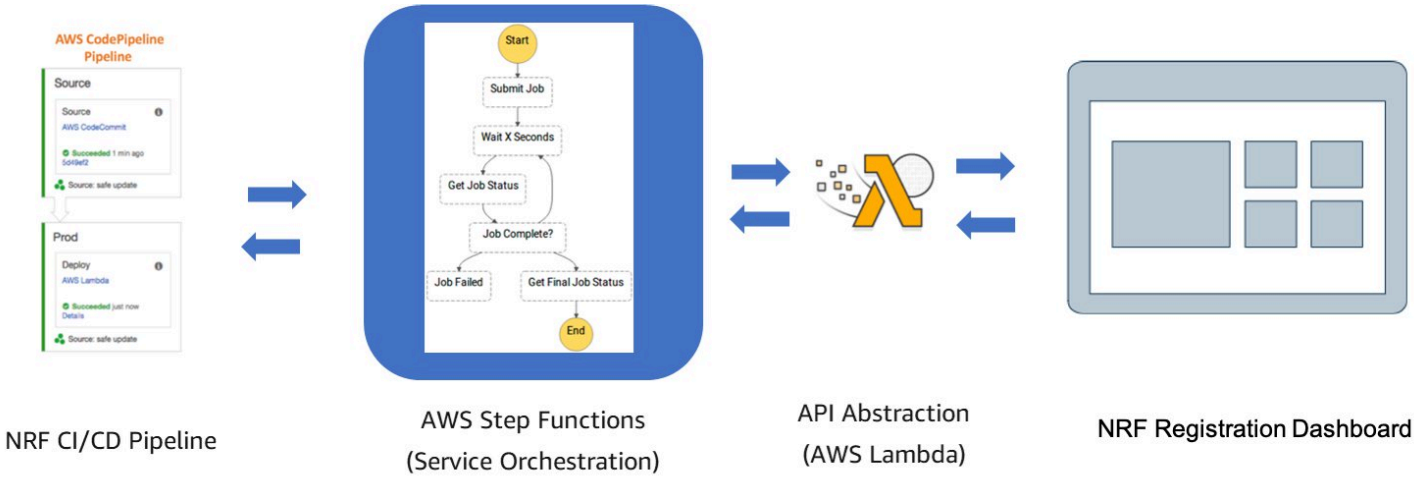
테스트

통신 관련 테스트 자동화 프레임워크는 코드 파이프라인에 통합할 수 있습니다. 코드 파이프라인은 Step Functions와 통합되어 테스트 자동화 프레임워크와의 통합을 오케스트레이션합니다. AWS Step Functions는 여러 Lambda 함수를 사용하여 API 호출을 통해 테스트 자동화 프레임워크(서드 파티 도구)를 호출합니다. Step Functions는 처음에 테스트 ID를 가져오고 테스트를 실행한 다음 테스트 자동화 프레임워크에서 결과를 가져옵니다. 그런 다음 결과를 분석하고 프로덕션 배포를 위한 애플리케이션의 승인을 위해 코드 파이프라인에 전달합니다. 승인은 필요에 따라 코드 파이프라인에서 자동화하거나 수동으로 유지 관리할 수 있습니다. 이는 CSP가 테스트 환경에서 배포를 프로덕션으로 승격하는 데 중요한 단계입니다. 통합에 필요한 상위 수준 API는 다음과 같은 방식으로 분류됩니다.

- 컨텍스트 가져오기
- 특정 테스트 케이스 실행
- 테스트 케이스 중지

- 결과 가져오기

외부 REST API 호출의 복잡성은 AWS Step Functions를 사용하여 모델링되며 이를 통해 표준 구문이 병렬 흐름을 호출하고, 결과를 기다리고, 조건에 따라 분기하고, REST API를 AWS CodePipeline과 통합할 수 있습니다.



테스트 흐름

CI/CD 및 오케스트레이션

지속적 통합 및 지속적 전달은 클라우드 네이티브 아키텍처와 함께 제공되는 전반적인 자동화 철학의 일부이며, 이를 5G에 적용하는 방식입니다. 오케스트레이션은 이 철학의 또 다른 측면으로, 네트워크에서 발생하는 모든 변경 사항에 동적으로 반응해야 합니다. 오케스트레이션과 CI/CD는 정상적인 서비스를 보장하고 서비스 중단을 최소화하기 위해 긴밀하게 결합되어야 합니다. CI/CD와 오케스트레이션의 통합은 두 가지 측면에서 이루어져야 합니다.

- 시스템에 패치와 업그레이드를 적용하는 작업은 라이브 서비스의 중단을 최소화하는 방식으로 관리하고 오케스트레이션해야 합니다. 예를 들어, 오케스트레이션은 업데이트를 배포할 최적의 시간을 동적으로 결정할 수 있습니다.
- CI/CD 인식 오케스트레이션은 도입된 배포 모델 전략(카나리, 선형 또는 모두 한 번에)에 따라 업그레이드를 배포하는 동안 트래픽 이동을 허용합니다.

일반적으로 오케스트레이션 솔루션은 CI/CD 파이프라인에서 실행되므로 오케스트레이션이 해당 파이프라인에 거버넌스 단계를 도입할 수 있으며 지속적인 업그레이드 주기에 노출될 수 있습니다.

결론

CI/CD는 개발자와 애플리케이션 팀이 단 몇 분 만에 새로운 애플리케이션 코드를 배포할 수 있는 명확하고 효율적인 경로를 제공합니다. AWS에는 AWS CodePipeline, AWS CodeCommit, AWS CodeBuild, AWS CodeDeploy 등을 포함하여 개발자가 새 코드를 통합, 테스트 및 배포하는 데 도움이 될 수 있는 다양한 도구가 있습니다. 이 문서에서는 새 코드 배포를 완료하는 데 필요한 여러 단계를 포함하여 AWS 서비스를 사용하여 5G 네트워크 기능을 완전히 자동화된 방식으로 배포하기 위한 CI/CD 프로세스를 생성하는 방법을 살펴보았습니다. 또한 서드 파티 테스트 자동화 프레임워크를 CI/CD 프로세스에 통합하고 Terraform과 같은 서드 파티 도구를 사용하는 방법도 살펴보았습니다.

기여자

이 문서를 작성하는 데 도움을 주신 분들입니다.

- Hisham Elshaer, Amazon Web Services AWS Telecom 부문 선임 컨설턴트
- Vara Prasad Talari, Amazon Web Services AWS Telecom 부문 수석 컨설턴트
- Rabi Abdel, Amazon Web Services AWS Telecom 부문 수석 컨설턴트
- Franco Bontorin, Amazon Web Services 공유 전달 부문 선임 컨설턴트
- Pragtideep Singh, Amazon Web Services 공유 전달 부문 컨설턴트
- Subbarao Duggisetty, Amazon Web Services Global Accounts 부문 클라우드 인프라 아키텍트
- 정영, Amazon Web Services AWS Telecom 선임 파트너 솔루션스 아키텍트

문서 개정

이 백서의 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하세요.

update-history-change	update-history-description	update-history-date
최초 게시	처음 게시된 백서	2021년 3월 8일

참조 문헌

다음에서 추가 정보를 참조하세요.

- Practicing Continuous Integration and Continuous Delivery on AWS(AWS에서 지속적 통합 및 지속적 전달 적용)(백서)
- Carrier-Grade Mobile Packet Core Network on AWS(AWS 기반의 통신사급 모바일 패킷 코어 네트워크)(백서)
- 5G Network Evolution with AWS(AWS를 기반으로 한 5G 네트워크 진화)(백서)

약어

- AMF - 액세스 및 이동성 관리 기능
- API - 애플리케이션 프로그래밍 인터페이스
- AUSF - 인증 서버 기능
- BSS - 비즈니스 지원 시스템
- CDK - Cloud Development Kit
- CI/CD - 지속적 통합 및 지속적 전달
- CLI - Command Line Interface
- CNF - 클라우드 네이티브 또는 컨테이너식 네트워크 기능
- CSP - 통신 서비스 제공업체
- CU - RAN 중앙 장치
- CVE - 일반적인 취약성 및 노출도
- DoS - 서비스 거부 공격
- DR - 재해 복구
- DU - RAN 분산 장치
- E2E - 엔드 투 엔드
- ECR - Elastic Container Registry
- EFS - Elastic File System
- EKS - Elastic Kubernetes Service
- EPC - Evolved Packet Core
- IaC - 코드형 인프라
- ISV - Independent Software Vendor
- MANO - 관리 및 오케스트레이션
- MEC - Multi-Access Edge Computing
- NACL - 네트워크 액세스 제어 목록
- NAT - 네트워크 주소 변환
- NF - 네트워크 기능
- NFV - 네트워크 기능 가상화
- NFVO - 네트워크 기능 가상화 오케스트레이터

- NOC - 네트워크 운영 센터
- NRF - 네트워크 리포지토리 기능
- OSS - 운영 지원 시스템
- PII - 개인 식별 정보
- QoS - 서비스 품질
- RAN - 무선 액세스 네트워크
- SBI - 서비스 기반 인터페이스
- SMF - 세션 관리 기능
- SSL - 보안 소켓 계층
- TaS - 서비스형 테스트
- TCP - Transmission Control Protocol
- TLS - 전송 계층 보안
- UDM - 통합 데이터 관리
- UDP - 사용자 데이터그램 프로토콜
- UPF - 사용자 영역 기능
- VIM - 가상화된 인프라 관리자
- VNF - 가상 네트워크 기능
- VPC - Virtual Private Cloud

고지 사항

고객은 본 문서에 포함된 정보를 독자적으로 평가할 책임이 있습니다. 본 문서는 (a) 정보 제공만을 위한 것이며, (b) 사전 고지 없이 변경될 수 있는 현재의 AWS 제품 제공 서비스 및 사례를 보여 주며, (c) AWS 및 자회사, 공급업체 또는 라이선스 제공자로부터 어떠한 약정 또는 보증도 하지 않습니다. AWS 제품 또는 서비스는 명시적이든 묵시적이든 어떠한 종류의 보증, 진술 또는 조건 없이 '있는 그대로' 제공됩니다. 고객에 대한 AWS의 책임과 법적 책임은 AWS 계약서에 준하며 본 문서는 AWS와 고객 간의 계약에 포함되지 않고 계약을 변경하지도 않습니다.