Amazon EKS 클러스터에 적합한 GitOps 도구 선택

AWS 권장 가이드

AWS 권장 가이드: Amazon EKS 클러스터에 적합한 GitOps 도구 선택

Table of Contents

Amazon EKS 클러스터에 적합한 GitOps 도구 선택

Pradip Kumar Pandey 및 Pratap Kumar Nanda, Amazon Web Services(AWS)

2025년 4월(문서 기록)

빠르게 진화하는 클라우드 네이티브 기술 환경에서 GitOps는 애플리케이션 및 인프라를 관리하고 배포하기 위한 강력한 방법론으로 부상했습니다. Amazon Elastic Kubernetes Service(Amazon EKS)를 사용하는 경우 GitOps 원칙을 구현하면 배포 프로세스를 크게 개선하고, 신뢰성을 개선하고, 운영을 간소화할 수 있습니다. 다양한 GitOps 도구를 사용할 수 있으며, EKS 클러스터에 적합한 도구를 선택하는 것은 팀의 효율성과 DevOps 관행의 전반적인 성공에 영향을 미칠 수 있는 중요한 결정입니다.

Amazon EKS 환경에 적합한 GitOps 도구를 선택하려면 특정 요구 사항, 팀 전문 지식, 확장성 요구 사항, 기존 와의 통합 기능 등 다양한 요소를 신중하게 고려해야 합니다 AWS 서비스. 각 도구에는 고유한 기능, 강점 및 잠재적 제한 사항이 있으므로 조직의 목표 및 운영 컨텍스트에 맞게 선택하는 것이 중요합니다.

이 가이드에서는 Amazon EKS용 GitOps 도구를 선택할 때 고려해야 할 주요 사항을 살펴보고, 자주 사용되는 옵션을 비교하고, 정보에 입각한 결정을 내리는 데 도움이 되는 인사이트를 제공합니다. 널리 사용되는 9가지 GitOps 도구를 다룹니다.

- Argo CD
- Flux
- 위브 GitOps
- Jenkins X
- GitLab CI/CD
- "
- 랜처 플릿
- Codefresh
- Pulumi

목표 비즈니스 성과

다음 목록은 개발 및 운영 프로세스에서 GitOps 원칙을 구현하는 도구를 선택할 때의 잠재적 목표와 결과에 대해 설명합니다.

Amazon EKS와의 원활한 통합

GitOps 도구는 Amazon EKS와 원활하게 통합되고 Amazon EKS 관련 기능 및 최적화와의 호환성을 제공해야 합니다.

- 기본 Amazon EKS 지원: 간편한 클러스터 연결 및 관리를 포함하여 Amazon EKS에 대한 기본 지원을 제공하는 도구를 찾습니다.
- AWS 서비스 통합: 도구가 AWS Identity and Access Management (IAM), Amazon Elastic Container Registry(Amazon ECR) 및 Amazon CloudWatch와 AWS 서비스 같은 다른와 상호 작용할 수 있는지 확인합니다.
- Amazon EKS 추가 기능 호환성: 도구가 Amazon EKS 추가 기능을 지원하고 이를 효과적으로 관리할 수 있는지 확인합니다.

확장성 및 성능

GitOps 도구는 소규모 클러스터에서 대규모 다중 클러스터 환경에 이르기까지 Amazon EKS 작업의 규모를 처리할 수 있어야 합니다.

- 리소스 효율성: 도구의 리소스 소비와 이것이 클러스터 성능에 미치는 영향을 평가합니다.
- 대규모 작업: 다양한 애플리케이션과 클러스터를 동시에 관리하는 도구의 기능을 평가합니다.
- 로드 시 성능: 고빈도 업데이트 및 대규모 배포 중에 도구의 성능을 고려합니다.

보안 및 규정 준수

보안 기능 및 규정 준수 기능은 특히 규제 대상 산업이나 민감한 데이터를 처리할 때 매우 중요합니다.

- 액세스 제어: IAM과 통합되는 강력한 역할 기반 액세스 제어(RBAC) 기능을 찾습니다.
- 보안 암호 관리: 도구가 민감한 정보를 처리하고 AWS Secrets Manager 또는 다른 솔루션과 통합하는 방법을 평가합니다.
- 감사 추적: 도구가 규정 준수 및 문제 해결을 위한 포괄적인 로깅 및 감사 기능을 제공하는지 확인합니다.
- 보안 스캔: 배포의 취약성에 대한 기본 제공 보안 스캔을 제공하는 도구를 고려합니다.

사용 편의성 및 학습 곡선

이 도구는 사용자 친화적이어야 하며 빠른 채택과 효율적인 사용을 보장하기 위해 팀의 기술에 부합해야 합니다.

- 사용자 인터페이스: 명령줄 인터페이스(CLI) 및 그래픽 사용자 인터페이스(GUI) 기능 모두의 직관적성을 평가합니다.
- 설명서 품질: 포괄적인 up-to-date 설명서 및 자습서를 찾습니다.
- 학습 리소스: 교육 자료, 과정 및 커뮤니티 리소스의 가용성을 고려합니다.

커뮤니티 및 네트워크 지원

강력한 커뮤니티와 네트워크는 귀중한 리소스, 플러그인 및 장기적인 지속 가능성을 제공할 수 있습니다.

- 활성 개발: 업데이트 빈도와 유지 관리자의 응답성을 확인합니다.
- 커뮤니티 크기: 지원 및 지식 공유를 위해 사용자 커뮤니티의 크기와 활동을 고려합니다.
- 타사 통합: 플러그인의 가용성과 스택의 다른 도구와의 통합을 평가합니다.

다중 클러스터 관리 기능

EKS 클러스터가 여러 개 있는 경우 클러스터를 효율적으로 관리하는 기능이 중요합니다.

- 중앙 집중식 관리: 단일 컨트롤 플레인에서 여러 클러스터를 관리할 수 있는 기능을 찾습니다.
- 클러스터 페더레이션: 다중 클러스터 애플리케이션을 위한 Kubernetes 페더레이션을 지원하는 도구를 고려합니다.
- 환경 패리티: 도구가 개발, 스테이징, 프로덕션과 같은 다양한 환경에서 일관성을 얼마나 잘 유지하는지 평가합니다.

관찰성 및 모니터링

도구는 배포 상태 및 클러스터 상태에 대한 명확한 인사이트를 제공해야 합니다.

- 배포 가시성: 배포 상태 및 기록에 대한 명확한 보기를 제공하는 기능을 찾습니다.
- 모니터링 도구와의 통합: 도구가 Prometheus 및 Grafana와 같은 널리 사용되는 모니터링 솔루션과 얼마나 잘 통합되는지 고려합니다.

- 알림 기능: 배포 문제 또는 드리프트에 대한 알림을 설정하고 관리하는 도구의 기능을 평가합니다.

유연성 및 사용자 지정

도구를 특정 워크플로 및 요구 사항에 맞게 조정하는 기능은 장기적인 만족을 위해 중요합니다.

- 확장성: 도구의 기능을 확장할 수 있는 플러그인 아키텍처 또는 APIs를 찾습니다.
- 사용자 지정 리소스 지원: 도구가 사용자 지정 Kubernetes 리소스를 효과적으로 처리할 수 있는지 확인합니다.
- 워크플로 사용자 지정: GitOps 워크플로를 팀의 필요에 맞게 얼마나 쉽게 조정할 수 있는지 평가합니다.

지속적인 제공 및 점진적 배포 지원

고급 배포 전략은 위험을 최소화하고 원활한 업데이트를 보장하는 데 매우 중요한 경우가 많습니다.

- 카나리아 배포: 카나리아 릴리스에 대한 기본 지원을 찾습니다.
- 블루/그린 배포: 블루/그린 배포 전략에 대한 도구의 기능을 평가합니다.
- 롤백 메커니즘: 장애가 발생한 배포에서 신속하게 복구easy-to-use 롤백 기능을 보장합니다.

비용 효율성 및 리소스 사용률

직접 및 간접 비용을 포함하여 도구를 채택하고 유지 관리하는 데 드는 전체 비용을 고려합니다.

- 라이선스 비용: 오픈 소스 옵션을 상용 솔루션과 비교하고 지원 및 엔터프라이즈 기능을 고려합니다.
- 운영 오버헤드: 관리 및 유지 관리 측면에서 추가 운영 비용을 평가합니다.
- 리소스 소비: 필요한 컴퓨팅 및 스토리지 리소스 측면에서 도구의 효율성을 평가합니다.

이러한 결과와 그 측면을 신중하게 고려하여 EKS 클러스터에 가장 적합한 GitOps 도구에 대해 정보에 입각한 결정을 내리고 도구가 조직의 요구 사항, 기능 및 장기 전략에 부합하는지 확인할 수 있습니다.

EKS 클러스터용 GitOps 도구

현재 시장에서 사용할 수 있는 Kubernetes용 GitOps 도구가 몇 가지 있습니다. 다음은 가장 널리 사용되는 몇 가지 옵션 목록입니다.

- Argo CD
- Flux
- 위브 GitOps
- Jenkins X
- GitLab CI/CD
- "
- 랜처 플릿
- Codefresh
- Pulumi

링크를 따라가면 이러한 도구가 GitOps 사례를 구현하는 방법에 대한 자세한 정보를 볼 수 있습니다. 각 도구에는 장점과 사용 사례가 있습니다. 선택은 특정 요구 사항, 기존 인프라, 팀 전문 지식 및 원하는 기능과 같은 요인에 따라 달라집니다. 조직의 요구 사항과 Kubernetes 환경의 복잡성을 기반으로 이러한 도구를 평가하는 것이 중요합니다.

Argo CD

Argo CD는 여러 주요 GitOps 원칙을 준수하는 Kubernetes에 널리 사용되는 GitOps 지속적 전달(CD) 도구입니다.

GitOps 지원

영역	도구 기능
선언적 구성	Argo CD는 Git 리포지토리에 저장된 선언적 구성을 사용합니다. 애플리케이션 및 인프라의 원하는 상태는 YAML 파일에 정의됩니다. 이러한 구성은 배포 방법이 아니라 배포해야 할 사항을 설명합니다.

영역	도구 기능
단일 정보 소스로서의 버전 제어 시스템	Git 리포지토리는 전체 시스템의 단일 정보 소스 역할을 합니다. 애플리케이션 및 인프라에 대한 모든 변경은 Git을 통해 이루어집니다. 이렇게 하면 전체 감사 추적과 이전 상태로 롤백할 수 있습니다.
자동 동기화	Argo CD는 Git 리포지토리의 변경 사항을 지속적으로 모니터링합니다. 변경 사항이 감지되면 클러스터의 실제 상태를 Git에 정의된 원하는 상태와 자동으로 동기화합니다. 이렇게 하면 클러스터가 항상 리포지토리에 설명된 상태를 반영합니다.
Kubernetes 네이티브	Argo CD는 Kubernetes 환경을 위해 특별히 설계되었습니다. 애플리케이션 관리를 위해 Kubernetes의 선언적 특성과 사용자 지정 리소스를 활용합니다.
자가 복구 및 드리프트 감지	Argo CD는 클러스터의 라이브 상태를 Git의 원하는 상태와 정기적으로 비교합니다. 드리프트(실제 상태와 원하는 상태 간의 차이)를 감지하면 이러한 불일치를 자동으로 수정할 수 있습니다.
다중 클러스터 및 다중 테넌시 지원	Argo CD는 단일 인스턴스에서 여러 Kubernetes 클러스터를 관리할 수 있습니다. 다중 테넌시를 지원하므로 팀마다 애플리케이션을 독립적으로 관리할 수 있습니다.
애플리케이션 정의	Argo CD의 애플리케이션은 Application CRD(사용자 지정 리소스 정의)를 사용하여 정의됩니다. 이를 통해 Kubernetes 네이티브 방식으로 배포 대상과 방법을 정의할 수 있습니다.

영역	도구 기능
배포 및 릴리스 분리	Argo CD는 코드 배포를 릴리스에서 사용자로 분리합니다. 이는 블루/그린 또는 카나리 배포와 같은 다양한 배포 전략을 통해 달성됩니다.
관찰성 및 감사 가능성	Argo CD는 애플리케이션 및 클러스터의 상태를 관찰하기 위한 웹 UI 및 CLI를 제공합니다. 모든 작업이 로깅되어 변경 사항 및 배포에 대한 명확한 감사 추적을 제공합니다.
보안 및 RBAC	Argo CD는 Kubernetes 역할 기반 액세스 제어(RBAC)와 통합됩니다. 인증 및 권한 부여를 위한 Single Sign-On 통합을 지원합니다.
플러그형 아키텍처	Argo CD는 다양한 소스 제어 관리 시스템, 차트 Helm, Kustomize 및 기타 Kubernetes 매니페스트 형식을 지원합니다. 이러한 유연성을 통해 다양한 환경과 워크플로에 맞출 수 있습니다.
지속적 전송(CD)	Argo CD는 지속적 전달에 중점을 두지만 지속적 통합(CI) 도구와 통합하여 완전한 CI/CD 파이프라인을 생성할 수 있습니다.

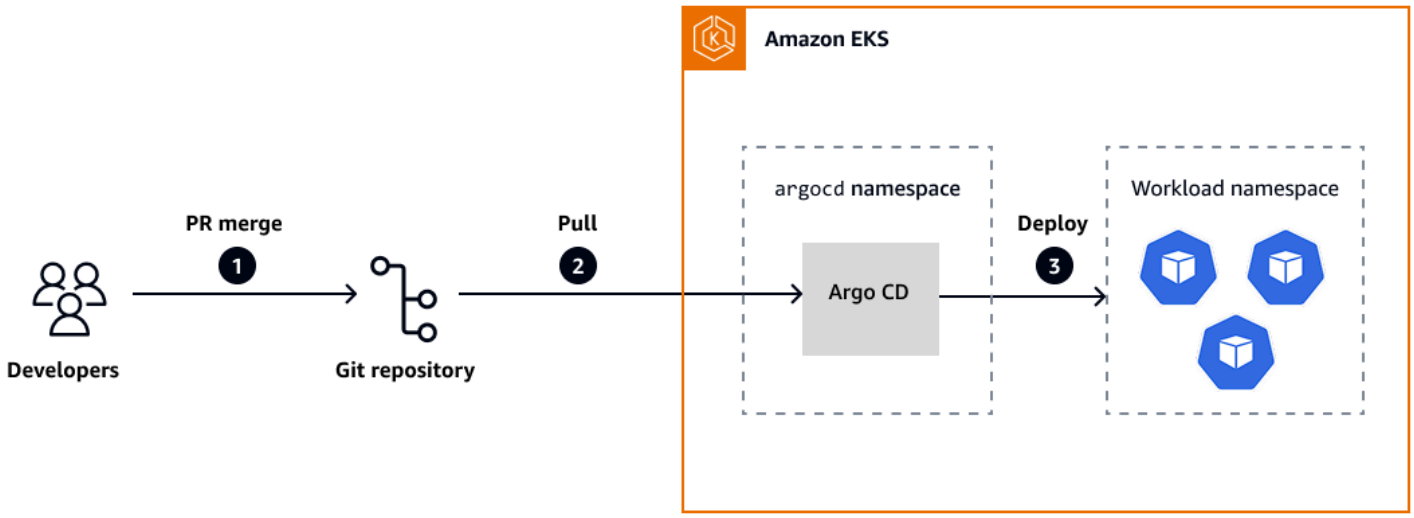
Argo CD는 이러한 GitOps 원칙을 준수함으로써 Kubernetes 배포를 관리할 수 있는 강력하고 확장 가능하며 안전한 방법을 제공합니다. 이를 통해 시스템의 운영 상태가 Git 리포지토리에 정의된 원하는 상태와 항상 동기화되고 복잡한 Kubernetes 환경에서 일관성, 신뢰성 및 관리 용이성을 높일 수 있습니다.

Argo CD가 해결할 수 있는 시나리오 및 요구 사항은이 가이드 뒷부분의 Argo CD 사용 사례를 참조하세요. Argo CD와 Flux 간의 비교는이 가이드 뒷부분의 기능 비교를 참조하세요.

자세한 내용은 Argo CD 설명서를 참조하세요.

아키텍처

다음 다이어그램은 EKS 클러스터 내에서 Argo CD를 사용하는 GitOps 기반 CD 워크플로를 보여줍니다. 자세한 내용은 Argo CD 설명서를 참조하세요.



여기서 각 항목은 다음과 같습니다.

- 1단계: 풀 요청(PR) 병합. 개발자는 Git 리포지토리에 저장된 Kubernetes 매니페스트 또는 차트 Helm에 변경 사항을 커밋합니다. PR을 검토하고 기본 브랜치에 병합하면 소스 제어에서 애플리케이션의 원하는 상태가 업데이트됩니다.
- 2단계: 리포지토리 동기화. Argo CD는 EKS 클러스터의 전용 네임스페이스(`argocd`) 내에서 실행되며 구성된 Git 리포지토리를 지속적으로 모니터링합니다. 변경 사항을 감지하면 최신 업데이트를 가져와 선언된 상태를 조정합니다.
- 3단계: 대상 네임스페이스에 배포. Argo CD는 Git의 원하는 상태를 클러스터의 라이브 상태와 비교합니다. 그런 다음 애플리케이션이 적절히 배포되거나 업데이트되도록 대상 워크로드 네임스페이스에 필요한 변경 사항을 적용합니다. 여기에는 배포, 서비스, ConfigMaps 및 보안 암호와 같은 Kubernetes 리소스를 관리하여 Git 소스와의 클러스터 일관성을 유지하는 것이 포함됩니다.

Flux

Flux는 GitOps 원칙을 고유한 방식으로 구현하는 Kubernetes용 또 다른 도구입니다.

GitOps 지원

영역	도구 기능
Git을 신뢰할 수 있는 단일 소스로 사용	Flux는 Git 리포지토리를 시스템의 원하는 상태를 정의하기 위한 최종 소스로 사용합니다. 애플

영역	도구 기능
	리케이션 및 인프라에 대한 모든 구성은 Git에 저장됩니다.
선언적 구성	Flux는 클러스터의 원하는 상태에 대한 선언적 설명과 함께 작동합니다. 이러한 설명은 일반적으로 Kubernetes 매니페스트, 차트 Helm 또는 Kustomize 오버레이입니다.
자동 비동기화	Flux는 Git 리포지토리의 변경 사항을 지속적으로 모니터링합니다. 변경 사항을 감지하면 클러스터에 자동으로 적용됩니다.
Kubernetes 네이티브	Flux는 Kubernetes 컨트롤러 및 사용자 지정 리소스 세트로 빌드됩니다. Kubernetes의 확장 메커니즘을 사용하여 GitOps 기능을 제공합니다.
풀 기반 배포 모델	기존 푸시 기반 CI/CD 시스템과 달리 Flux는 풀 기반 모델을 사용합니다. 클러스터는 외부 시스템을 사용하여 변경 사항을 푸시하는 대신 Git에서 원하는 상태를 가져옵니다.
지속적 조정	Flux는 클러스터의 실제 상태를 Git의 원하는 상태와 지속적으로 비교합니다. 이러한 상태 간에 감지된 모든 드리프트를 자동으로 수정합니다.
멀티테넌시	Flux는 Kustomizations 및 HelmReleases 개념을 통해 다중 테넌시를 지원합니다. 팀마다 구성의 자체 부분을 독립적으로 관리할 수 있습니다.
점진적 전달	Flux는 Flagger 구성 요소를 통해 canary 릴리스 및 A/B 테스트와 같은 고급 배포 전략을 지원합니다.
Helm 통합	Flux에는 Helm에 대한 기본 지원이 포함되어 있으므로 GitOps를 통해 Helm 릴리스를 쉽게 관리할 수 있습니다.

영역	도구 기능
이미지 업데이트 자동화	Flux는 컨테이너 레지스트리에서 새 버전을 사용할 수 있을 때 Git에서 컨테이너 이미지를 자동으로 업데이트할 수 있습니다.
Kustomize 지원	Flux for Kustomize에서 제공하는 기본 지원을 사용하여 Kubernetes 매니페스트를 사용자 지정하고 패치할 수 있습니다.
보안 및 RBAC	Flux는 액세스 제어를 위해 Kubernetes RBAC와 통합됩니다. 다양한 백엔드를 통한 보안 암호 관리를 지원합니다.
관찰성	Flux는 조정 및 작업에 대한 상태 정보와 지표를 제공합니다. 향상된 관찰성을 위해 모니터링 도구와 통합됩니다.
이벤트 중심 아키텍처	Flux는 이벤트 기반 접근 방식을 사용하여 조정 및 업데이트를 구현합니다.
확장성	이 도구는 확장 가능하도록 설계되었으므로 사용자 지정 컨트롤러와 리소스를 추가할 수 있습니다.
클러스터 간 동기화	Flux는 단일 리포지토리 세트에서 여러 클러스터의 관리를 지원합니다.
종속성 관리	이를 통해 시스템의 여러 부분 간에 종속성을 정의할 수 있으며 올바른 작업 순서를 보장할 수 있습니다.
Webhook 수신기	Git 공급자 또는 기타 시스템으로부터 웹후크를 수신하여 즉시 조정을 시작하도록 Flux를 구성할 수 있습니다.

Flux는 이러한 GitOps 원칙을 구현하여 Kubernetes 클러스터 및 애플리케이션을 관리하기 위한 강력하고 유연한 시스템을 제공합니다. 이를 통해 인프라 및 애플리케이션이 항상 Git 리포지토리와 동기

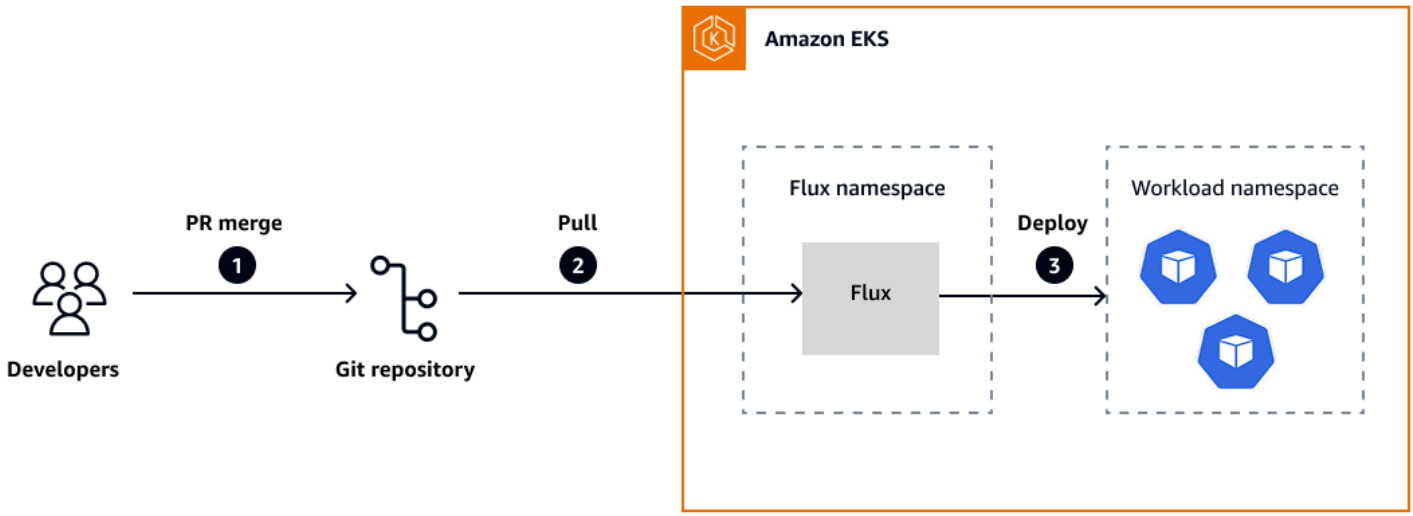
화되고 복잡한 Kubernetes 환경에서 일관성, 안정성 및 관리 용이성을 제공할 수 있습니다. 이 도구의 Kubernetes 네이티브 접근 방식과 자동화에 초점을 맞추면 클라우드 네이티브 환경에 특히 적합합니다.

Flux가 해결할 수 있는 시나리오 및 요구 사항은이 가이드 뒷부분의 Flux 사용 사례를 참조하세요. Argo CD와 Flux 간의 비교는이 가이드 뒷부분의 기능 비교를 참조하세요.

자세한 내용은 Flux 설명서를 참조하세요.

아키텍처

다음 다이어그램은 EKS 클러스터 내에서 Flux를 사용하는 GitOps 기반 CD 워크플로를 보여줍니다. 자세한 내용은 Flux 설명서를 참조하세요.



여기서 각 항목은 다음과 같습니다.

- 1단계: 풀 요청(PR) 병합. 개발자는 Git 리포지토리에 저장된 Kubernetes 매니페스트 또는 차트 Helm에 변경 사항을 커밋합니다. PR을 검토하고 기본 브랜치에 병합하면 소스 제어에서 애플리케이션의 원하는 상태가 업데이트됩니다.
- 2단계: 리포지토리 동기화. Flux는 EKS 클러스터의 전용 네임스페이스 내에서 실행되며 구성된 Git 리포지토리를 지속적으로 모니터링합니다. 변경 사항을 감지하면 최신 업데이트를 가져와 선언된 상태를 조정합니다.
- 3단계: 대상 네임스페이스에 배포. Flux는 Git의 원하는 상태를 클러스터의 라이브 상태와 비교합니다. 그런 다음 애플리케이션이 적절히 배포되거나 업데이트되도록 대상 워크로드 네임스페이스에 필요한 변경 사항을 적용합니다.

위브 GitOps

Weave GitOps는 GitOps라는 용어를 도입한 회사인 Weaveworks에서 개발했습니다. 이 도구는 핵심 GitOps 원칙을 기반으로 하는 포괄적인 GitOps 솔루션을 제공합니다.

GitOps 지원

영역	도구 기능
Git을 신뢰할 수 있는 단일 소스로 사용	Weave GitOps는 Git 리포지토리를 시스템의 원하는 상태를 정의하기 위한 신뢰할 수 있는 소스로 사용합니다. 애플리케이션 매니페스트, 인프라 정의 및 정책을 포함한 모든 구성은 Git에 저장됩니다.
선언적 구성	시스템은 전체 시스템 상태에 대한 선언적 설명을 사용합니다. 이러한 설명은 일반적으로 Kubernetes 매니페스트, 차트 Helm 또는 기타 선언적 형식입니다.
자동 동기화	Weave GitOps는 Git 리포지토리의 변경 사항을 지속적으로 모니터링합니다. 변경 사항을 감지하면 대상 환경에 자동으로 적용됩니다.
Kubernetes 네이티브 아키텍처	Weave GitOps는 Kubernetes 컨트롤러 및 사용자 지정 리소스 세트로 빌드됩니다. Kubernetes의 확장 메커니즘을 사용하여 GitOps 기능을 제공합니다.
지속적 조정	이 도구는 클러스터의 실제 상태를 Git에 정의된 원하는 상태와 지속적으로 비교합니다. 이러한 상태 간에 감지된 모든 드리프트를 자동으로 수정합니다.
다중 클러스터 관리	Weave GitOps는 단일 컨트롤 플레인에서 여러 Kubernetes 클러스터의 관리를 지원합니다. 이를 통해 다양한 환경에 일관되게 애플리케이션을 배포할 수 있습니다.

영역	도구 기능
코드형 정책	Weave GitOps는 보안 및 규정 준수 규칙을 적용하기 위한 코드로 정책의 개념을 통합합니다. 정책은 애플리케이션 코드 및 인프라 정의와 함께 버전 관리됩니다.
점진적 전달	이 도구는 canary 릴리스 및 블루/그린 배포와 같은 고급 배포 전략을 지원합니다. 자동적이고 점진적인 전송을 위해 Flagger와 통합됩니다.
관찰성 및 대시보드	Weave GitOps는 애플리케이션 및 클러스터의 상태를 모니터링하기 위한 기본 제공 대시보드를 제공합니다. 조정 프로세스 및 클러스터 상태에 대한 인사이트를 제공합니다.
설계에 따른 보안	이 도구는 RBAC 통합 및 보안 암호 관리를 비롯한 보안 모범 사례를 구현합니다. 다양한 인증 방법을 지원하고 엔터프라이즈 자격 증명 공급자와 통합됩니다.
확장성 및 통합	이 도구는 다양한 클라우드 네이티브 도구와 함께 작동하도록 설계되었습니다. Flux, Helm, Kustomize와 같은 인기 있는 도구를 지원합니다.
셀프 서비스 개발자 플랫폼	Weave GitOps를 사용하면 개발자를 위한 셀프 서비스 플랫폼을 생성할 수 있습니다. 애플리케이션 배포를 위한 템플릿과 가드레일을 제공합니다.
GitOps 자동화	이 도구는 업데이트를 위한 풀 요청 생성을 포함하여 GitOps 워크플로의 여러 측면을 자동화합니다.
지속적 전송 파이프라인	CI/CD 시스템과 통합되어 end-to-end 전송 파이프라인을 생성합니다.

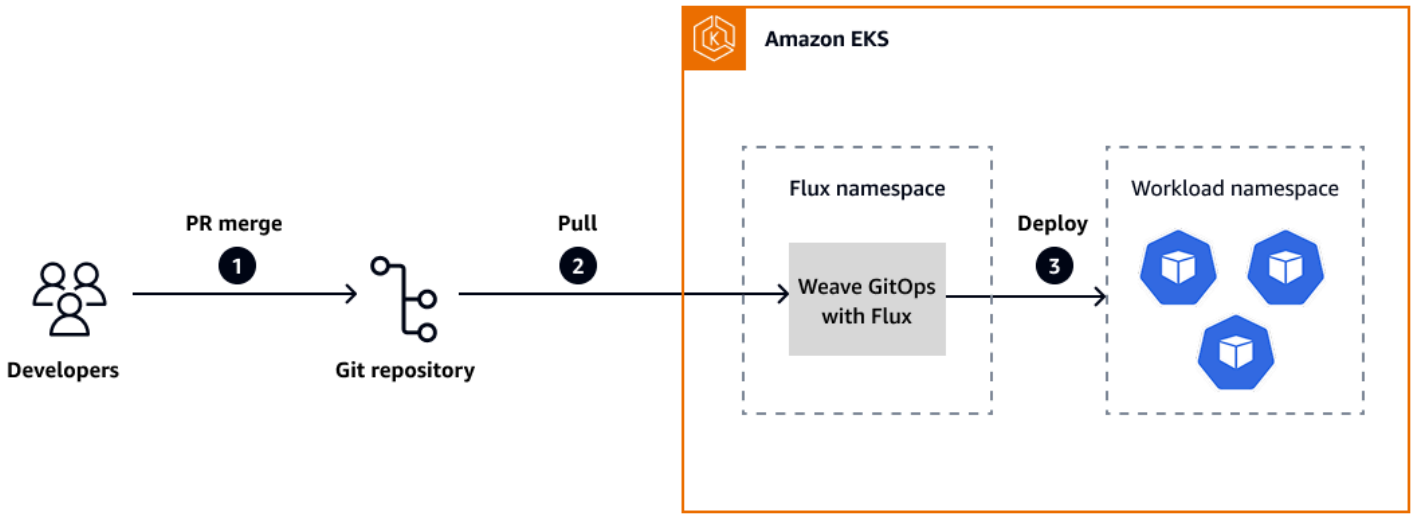
영역	도구 기능
감사 및 규정 준수	Weave DevOps는 모든 변경 사항 및 작업에 대한 전체 감사 추적을 제공합니다. 버전 관리 및 자동화된 프로세스를 통해 규정 준수 요구 사항을 충족하는 데 도움이 됩니다.
확장성	이 도구는 소규모 프로젝트에서 대규모 엔터프라이즈급 배포로 확장하도록 설계되었습니다.
팀 공동 작업	Weave GitOps는 Git 기반 워크플로를 통해 개발 팀과 운영 팀 간의 협업을 용이하게 합니다.
서비스형 GitOps	이 도구는 GitOps를 관리형 서비스로 제공하여 채택 및 관리를 간소화합니다.
하이브리드 및 멀티 클라우드 지원	Weave GitOps를 사용하면 다양한 클라우드 공급자와 온프레미스 환경에서 일관된 관리를 수행할 수 있습니다.
지속적인 보안	이 도구는 배포 프로세스 전반에 걸쳐 보안 스캔 및 정책 적용을 통합합니다.

Weave GitOps는 이러한 원칙을 구현하여 기본 배포 자동화를 넘어 포괄적인 GitOps 솔루션을 제공합니다. 보안, 확장성 및 사용 편의성에 중점을 둔 클라우드 네이티브 애플리케이션을 위한 완전한 운영 모델을 만드는 것을 목표로 합니다. 이러한 GitOps 원칙을 준수함으로써 Weave GitOps는 조직이 여러 클러스터 및 클라우드 공급자에서 Kubernetes 환경을 일관되고 감사 가능하며 효율적으로 관리할 수 있도록 지원합니다.

자세한 내용은 Weave GitOps 설명서를 참조하세요.

아키텍처

다음 다이어그램은 EKS 클러스터 내에서 Weave GitOps를 사용하는 GitOps 기반 CD 워크플로를 보여줍니다. 자세한 내용은 Weave GitOps 리포지토리를 참조하세요.



여기서 각 항목은 다음과 같습니다.

- 1단계: 풀 요청(PR) 병합. 개발자는 Git 리포지토리에 저장된 Kubernetes 매니페스트 또는 차트 Helm에 변경 사항을 커밋합니다. PR을 검토하고 기본 브랜치에 병합하면 소스 제어에서 애플리케이션의 원하는 상태가 업데이트됩니다.
- 2단계: 리포지토리 동기화. Weave GitOps는 EKS 클러스터의 Flux 네임스페이스 내에서 실행되며 구성된 Git 리포지토리를 지속적으로 모니터링합니다. 변경 사항을 감지하면 최신 업데이트를 가져와 선언된 상태를 조정합니다.
- 3단계: 대상 네임스페이스에 배포. 위브 GitOps는 Git의 원하는 상태를 클러스터의 라이브 상태와 비교합니다. 그런 다음 애플리케이션이 적절히 배포되거나 업데이트되도록 대상 워크로드 네임스페이스에 필요한 변경 사항을 적용합니다.

Jenkins X

Jenkins X는 Kubernetes 환경에 대한 GitOps 원칙을 구현하는 클라우드 네이티브 오픈 소스 CI/CD 플랫폼입니다. Jenkins X는 Argo CD 또는 Flux와 같은 GitOps 도구만은 아니지만 GitOps 관행을 워크플로에 통합합니다.

GitOps 지원

영역	도구 기능
Git 중심 워크플로	Jenkins X는 Git 리포지토리를 애플리케이션 코드와 구성 모두에 대한 기본 정보 소스로 사용합

영역	도구 기능
	니다. 애플리케이션 및 인프라에 대한 모든 변경은 Git을 통해 이루어집니다.
코드형 환경(EaC)	환경(예: 스테이징 및 프로덕션)은 Git 리포지토리에서 코드로 정의됩니다. 이를 통해 버전 관리 및 환경 구성 검토가 가능합니다.
자동화된 CI/CD 파이프라인	Jenkins X는 프로젝트에 대한 CI/CD 파이프라인을 자동으로 설정합니다. 이러한 파이프라인은 코드(파이프라인 코드)로 정의되고 Git에 저장됩니다.
Kubernetes 네이티브	Jenkins X는 Kubernetes 환경을 위해 특별히 구축되었습니다. Kubernetes 리소스와 사용자 지정 리소스 정의(CRDs 사용합니다.
미리 보기 환경	Jenkins X는 풀 요청에 대한 임시 환경을 자동으로 생성합니다. 병합 전에 변경 사항을 쉽게 검토하고 테스트할 수 있습니다.
환경 간 홍보	Jenkins X는 GitOps 접근 방식을 사용하여 환경 간에 애플리케이션을 홍보합니다(예: 스테이징에서 프로덕션으로). 적절한 검토 및 승인 프로세스를 보장하기 위해 풀 요청을 사용하여 승격을 처리합니다.
차트 Helm 관리	Jenkins X는 Helm 차트를 사용하여 애플리케이션을 패키징하고 배포합니다. 차트는 Git 리포지토리에서 버전 관리됩니다.
자동 버전 관리	Jenkins X는 애플리케이션 및 릴리스의 버전 관리를 자동으로 관리합니다. 의미 체계 버전 관리를 사용하고 릴리스 정보를 생성합니다.
ChatOps 통합	Jenkins X는 일반적인 작업을 위해 ChatOps를 지원합니다. 이는 자동화 및 협업의 GitOps 원칙에 부합합니다.

영역	도구 기능
확장성	이 도구는 기능을 확장하기 위한 플러그인 시스템을 제공합니다. 이를 통해 다양한 클라우드 네이티브 도구와 통합할 수 있습니다.
코드형 인프라(IaC)	Jenkins X는 인프라를 정의하고 관리하기 위해 Terraform AWS Cloud Development Kit (AWS CDK), CloudFormation, 및 기타 IaC 도구를 지원합니다. 인프라 정의는 애플리케이션 코드와 함께 버전이 제어됩니다.
자동 롤백	Jenkins X는 배포 후 문제가 감지되면 자동 롤백을 지원합니다.
보안 암호 관리	이 도구는 외부 보안 암호 관리 솔루션과 통합되어 민감한 정보를 안전하게 처리합니다.
관찰성	Jenkins X는 관찰성을 위한 모니터링 및 로깅 도구와의 통합을 제공합니다.
다중 클라우드 지원	Jenkins X는 다양한 클라우드 공급자와 온프레미스 환경에서 작동하도록 설계되었습니다.
팀 공동 작업	이 도구는 Git 기반 워크플로 및 풀 요청을 통해 협업을 장려합니다.
지속적인 피드백	이 도구는 자동화된 테스트 및 미리 보기 환경을 통해 변경 사항에 대한 빠른 피드백을 제공합니다.
DevOps 모범 사례	Jenkins X는 기본적으로 GitOps 원칙을 포함한 DevOps 모범 사례를 구현합니다. GitOps
선언적 구성	이 도구는 선언적 구성을 사용하여 애플리케이션 및 환경을 정의합니다.
자동 업그레이드	Jenkins X는 Jenkins X 플랫폼 자체의 업그레이드를 자동화하는 도구를 제공합니다.

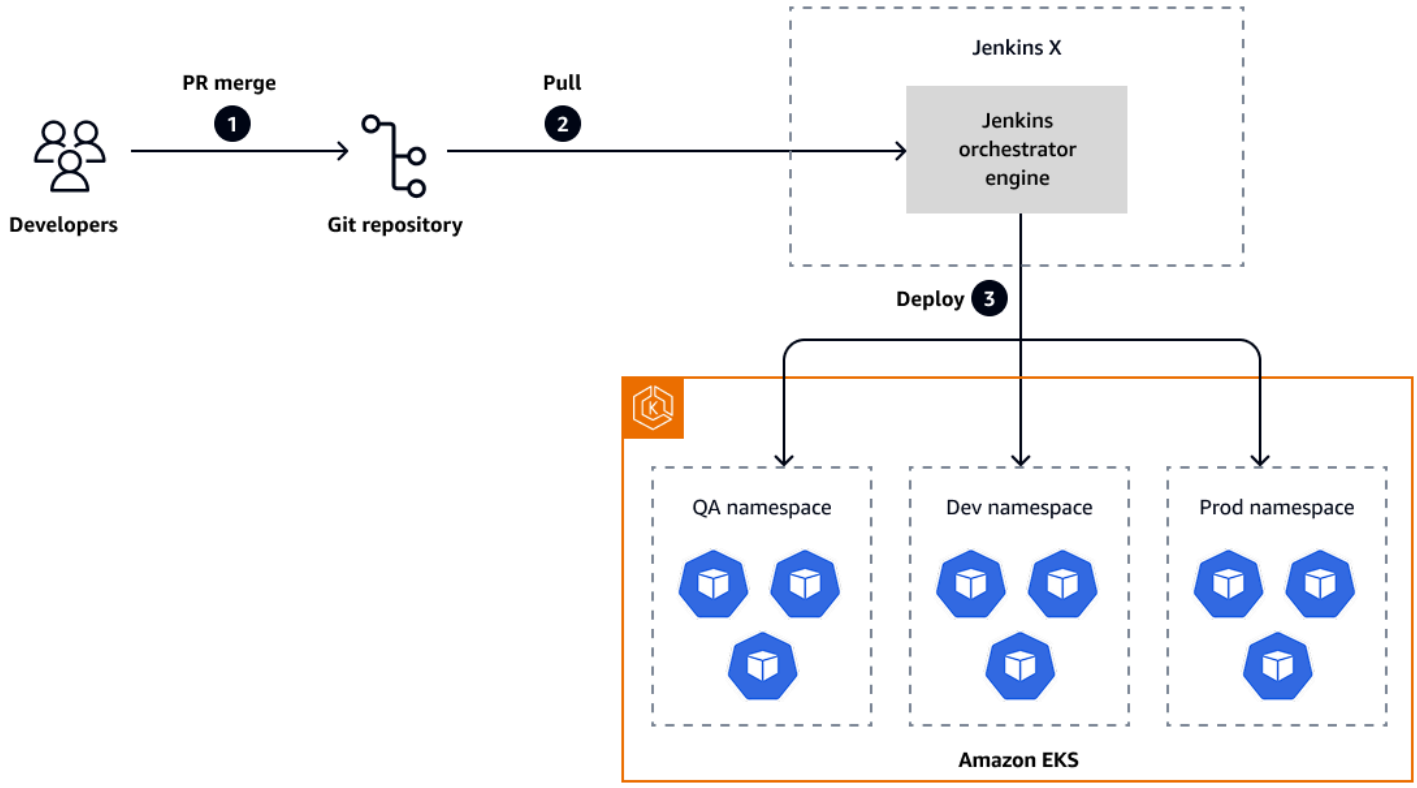
Jenkins X는 이러한 GitOps 원칙을 구현하여 Kubernetes에 대한 포괄적인 CI/CD 솔루션을 생성합니다. 이는 GitOps 관행을 준수하면서 코드 커밋에서 프로덕션 배포에 이르기까지 전체 소프트웨어 제공 프로세스를 자동화하고 간소화하는 것을 목표로 합니다. 이렇게 하면 팀이 클라우드 네이티브 환경에서 더 빠르고 안정적이며 일관된 배포를 달성할 수 있습니다.

Jenkins X와 Argo CD 또는 Flux와 같은 도구의 주요 차이점은 Jenkins X가 배포 및 환경 관리를 위한 GitOps 원칙을 통합하면서 빌드 자동화 및 파이프라인 관리를 포함한 보다 포괄적인 CI/CD 솔루션을 제공한다는 것입니다. 따라서 단일 GitOps 프레임워크 내에서 CI 및 CD 측면을 모두 다루는 all-in-one 솔루션이 필요한 팀에 특히 적합합니다.

자세한 내용은 Jenkins X 설명서를 참조하세요.

아키텍처

다음 다이어그램은 Jenkins X를 사용하는 GitOps 기반 CD 워크플로를 보여줍니다. 자세한 내용은 Jenkins X 설명서를 참조하세요.



여기서 각 항목은 다음과 같습니다.

- 1단계: 풀 요청(PR) 병합. 개발자는 Kubernetes 매니페스트, 차트 Helm 또는 Git 리포지토리에 저장된 애플리케이션 코드에 대한 변경 사항을 포함하는 풀 요청을 생성합니다. 검토 및 승인 후 PR이 기본 브랜치에 병합되고 소스 제어에서 원하는 상태가 업데이트됩니다.

- 2단계: 리포지토리 동기화. Jenkins X는 변경 사항을 감지하면 CI/CD 파이프라인을 자동으로 트리거합니다. 파이프라인은 GitOps 원칙을 사용하여 다양한 환경(예: 스테이징 및 프로덕션)을 통해 애플리케이션을 빌드, 테스트 및 홍보합니다.
- 3단계: 대상 네임스페이스에 배포. Jenkins X는 환경 리포지토리(스테이징 및 프로덕션)를 새 애플리케이션 버전으로 업데이트합니다. 클러스터는 Git에서 최신 매니페스트를 가져와 애플리케이션을 적절한 네임스페이스에 배포하여 변경 사항을 자동으로 조정합니다.

GitLab CI/CD

GitLab CI/CD는 지속적인 통합, 전송 및 배포 기능을 제공하는 GitLab 플랫폼의 통합 부분입니다. GitLab CI/CD는 GitOps 도구만은 아니지만 특히 Kubernetes 배포에 사용할 때 GitOps 원칙을 구현하도록 구성할 수 있습니다.

GitOps 지원

영역	도구 기능
Git을 신뢰할 수 있는 단일 소스로 사용	GitLab CI/CD는 Git 리포지토리를 사용하여 애플리케이션 코드와 인프라 구성을 모두 저장합니다. 시스템에 대한 모든 변경 사항은 Git을 통해 이루어지므로 전체 기록 및 감사 추적이 보장됩니다.
선언적 구성	GitLab CI/CD 파이프라인은 Git 리포지토리에 저장된 선언적 구성인 .gitlab-ci.yml 파일에 정의됩니다. Kubernetes 매니페스트, 차트 Helm 또는 기타 코드형 인프라(IaC) 파일을 동일한 리포지토리에 저장하여 원하는 인프라 상태를 정의할 수 있습니다.
자동화된 파이프라인	GitLab CI/CD는 변경 사항이 리포지토리로 푸시될 때 파이프라인을 자동으로 트리거합니다. 이러한 파이프라인에는 애플리케이션 구축, 테스트 및 배포 단계가 포함될 수 있습니다.
Kubernetes 통합	GitLab CI/CD는 네이티브 Kubernetes 통합을 제공하고 Kubernetes 클러스터에 대한 GitOps 스

영역	도구 기능
	타일 배포를 지원합니다. Git의 구성을 기반으로 Kubernetes 리소스를 자동으로 생성하고 관리할 수 있습니다.
환경 관리	GitLab CI/CD는 여러 환경(예: 스테이징 및 프로덕션)의 정의를 코드로 지원합니다. 이러한 환경에 대한 배포는 GitOps 관행에 따라 자동화되거나 수동 승인이 필요할 수 있습니다.
애플리케이션 검토	GitLab은 다른 GitOps 도구의 미리 보기 환경과 마찬가지로 병합 요청을 위한 임시 환경을 자동으로 생성할 수 있습니다. 이를 통해 병합 전에 변경 사항을 쉽게 검토하고 테스트할 수 있습니다.
지속적 배포	GitLab CI/CD는 변경 사항이 특정 브랜치에 병합될 때 Kubernetes 클러스터에 변경 사항을 자동으로 배포하도록 구성할 수 있습니다.
IaC	GitLab CI/CD는 Terraform 및와 같은 도구와의 통합을 지원 CloudFormation 하여 인프라를 코드로 관리합니다. 인프라 정의는 애플리케이션 코드와 함께 버전을 제어할 수 있습니다.
관찰성 및 모니터링	GitLab CI/CD는 Prometheus 및 Grafana와의 통합을 포함하여 내장된 모니터링 및 관찰성 기능을 제공합니다.
보안 스캔	GitLab CI/CD에는 CI/CD 파이프라인에 통합하여 GitOps 워크플로의 일부로 보안을 적용할 수 있는 보안 스캔 도구가 내장되어 있습니다.
컨테이너 레지스트리	GitLab CI/CD에는 GitOps 워크플로에서 컨테이너 이미지 관리를 원활하게 통합하기 위한 컨테이너 레지스트리가 내장되어 있습니다.

영역	도구 기능
Auto DevOps	GitLab CI/CD의 Auto DevOps 기능은 Kubernetes 배포에 대한 GitOps 원칙을 따르는 CI/CD 파이프라인을 자동으로 구성할 수 있습니다.
승인 워크플로	GitLab CI/CD는 환경 간에 제어된 프로모션을 제공하는 배포에 대한 승인 프로세스를 지원합니다.
보안 암호 관리	GitLab CI/CD는 CI/CD 파이프라인 내에서 보안 암호를 안전하게 관리하고 사용하는 기능을 제공합니다.
버전 관리 및 릴리스	GitLab CI/CD는 CI/CD 프로세스의 일부로 자동 버전 관리 및 릴리스 관리를 지원합니다.
롤백	GitLab CI/CD를 사용하면 배포 후 문제가 감지되면 이전 버전으로 쉽게 롤백할 수 있습니다.
감사 로그	GitLab CI/CD는 GitOps의 추적 가능성 측면을 지원하기 위해 모든 작업에 대한 포괄적인 감사 로그를 제공합니다.
다중 프로젝트 파이프라인	GitLab CI/CD는 여러 프로젝트 또는 리포지토리에 걸쳐 있는 복잡한 GitOps 워크플로를 지원합니다.
ChatOps	GitLab CI/CD는 채팅 인터페이스를 통해 협업 및 작업을 제공하는 ChatOps 통합을 지원합니다.
Kubernetes 클러스터 관리	GitLab CI/CD는 GitLab 인터페이스에서 직접 Kubernetes 클러스터를 관리하는 기능을 제공합니다.

GitLab CI/CD는 GitOps용으로 독점적으로 설계되지는 않았지만, 특히 이미 GitLab을 기본 개발 플랫폼으로 사용하는 팀의 경우 GitOps 관행을 구현하는 데 효과적으로 사용할 수 있습니다. GitLab 소스 제어, CI/CD 및 Kubernetes 관리를 결합한 통합 접근 방식을 사용하면 GitOps 워크플로를 구현하기 위한 강력한 도구가 됩니다.

GitLab CI/CD와 Argo CD 또는 Flux와 같은 전용 GitOps 도구 간의 주요 차이점은 GitLab이 CI/CD 기능과 함께 소스 제어 관리, 문제 추적 및 기타 개발 도구를 포함하는 보다 포괄적인 플랫폼을 제공한다는 것입니다. 따라서 더 광범위한 개발 시스템 내에서 GitOps 관행을 구현할 수 있는 all-in-one 솔루션이 필요한 팀에 특히 적합합니다.

GitLab CI/CD 및 아키텍처에 대한 자세한 내용은 GitLab CI/CD 설명서를 참조하세요.

"

GitOps 도구로만 설계된 것은 아니지만 특히 클라우드 네이티브 및 Kubernetes 배포에 사용할 때 GitOps 원칙을 구현하도록 구성할 수 있습니다.

GitOps 지원

영역	도구 기능
선언적 구성	일반적으로 JSON 또는 YAML 파일로 저장되는 선언적 파이프라인 정의를 사용합니다. 이러한 파이프라인 정의는 GitOps 관행에 따라 Git 리포지토리에서 버전을 제어할 수 있습니다.
IaC	코드형 인프라 및 배포 구성의 정의를 지원합니다. 이러한 정의는 Git 리포지토리에 저장할 수 있으며 단일 정보 소스 역할을 할 수 있습니다.
다중 클라우드 배포	SDK는 여러 클라우드 공급자 및 Kubernetes 클러스터에서 작동하도록 설계되었습니다. 이를 통해 다양한 환경에서 일관된 GitOps 관행을 구현할 수 있습니다.
파이프라인을 코드로	" 파이프라인은 코드로 정의하여 Git 리포지토리에 저장할 수 있습니다. 이를 통해 버전 관리 및 배포 프로세스 검토가 가능합니다.

영역	도구 기능
자동 배포	Git 리포지토리의 변경 사항에 따라 배포를 자동으로 시작하도록 Mount를 구성할 수 있습니다. 이 도구는 GitOps의 핵심인 지속적인 배포 사례를 지원합니다.
변경 불가능한 인프라	GitOps의 주요 개념인 변경 불가능한 인프라 사용을 촉진합니다. 기존 인스턴스를 수정하는 대신 새 인스턴스를 배포하는 것이 좋습니다.
롤백 및 버전 관리	강력한 롤백 기능과 이전에 알려진 정상 상태로의 빠른 되돌리기를 제공합니다. 추적성의 GitOps 원칙에 따라 배포 버전 관리를 지원합니다.
승인 워크플로	환경 간 제어된 프로모션을 지원하기 위해 파이프라인에 수동 판단 단계를 포함합니다. 이는 배포와 릴리스 간의 분리에 대한 GitOps 사례를 지원합니다.
카나리아 및 블루/그린 배포	JD는 안전하고 제어된 릴리스에 대한 GitOps 관행에 부합하는 고급 배포 전략을 지원합니다.
버전 관리 시스템과의 통합	또한 다양한 Git 공급자와 통합하여 리포지토리 이벤트를 기반으로 파이프라인을 시작할 수 있습니다.
Kubernetes 통합	Kubernetes에 대한 기본 지원을 제공하고 Kubernetes 리소스의 GitOps 스타일 관리를 지원합니다.
아티팩트 관리	JD는 GitOps 워크플로를 유지 관리하는 데 중요한 아티팩트 관리 및 버전 관리를 지원합니다.
관찰성 및 모니터링	SDK는 GitOps의 관찰성 측면을 지원하기 위해 모니터링 도구와의 통합을 제공합니다.

영역	도구 기능
감사 추적	SDK는 GitOps의 감사 원칙을 지원하는 자세한 배포 로그 및 기록을 제공합니다.
역할 기반 액세스 제어(RBAC)	이 도구는 GitOps 보안 관행에 따라 누가 어떤 작업을 수행할 수 있는지에 대한 세분화된 제어를 위해 RBAC를 구현합니다.
템플릿 지정 및 파라미터화	재사용 가능하고 파라미터화된 배포를 활성화하기 위해 파이프라인 정의에서 템플릿 지정을 지원합니다.
환경 홍보	통제된 방식으로 환경 간(예: 스테이징에서 프로덕션으로) 애플리케이션 홍보를 용이하게 합니다.
CI 도구와의 통합	다양한 지속적 통합(CI) 도구와 통합하여 GitOps 원칙을 준수하는 완전한 CI/CD 파이프라인을 제공할 수 있습니다.
사용자 지정 단계 및 확장	이 도구는 사용자 지정 단계 및 확장을 지원하므로 팀은 필요에 맞는 GitOps 워크플로를 구현할 수 있습니다.
중앙 집중식 관리	Oracle은 여러 환경 및 클라우드 공급자에 대한 배포를 관리하기 위한 중앙 집중식 플랫폼을 제공합니다.

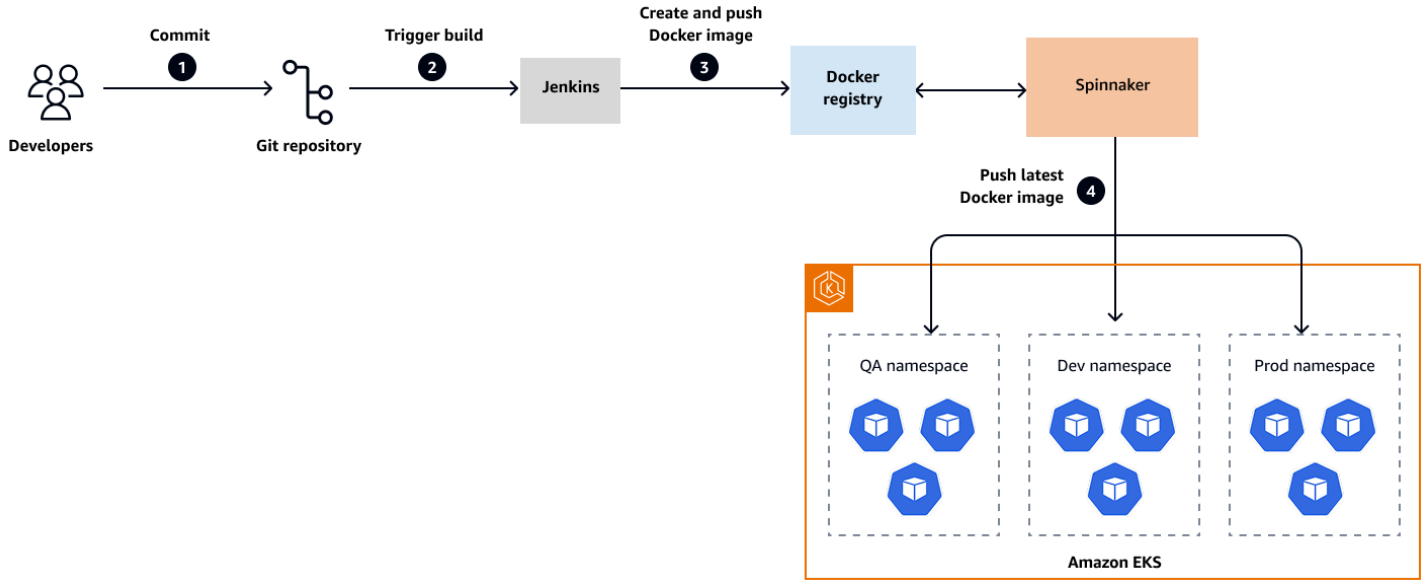
주로 GitOps 도구로 판매되지는 않지만 유연성과 강력한 기능 세트를 통해 특히 복잡한 멀티 클라우드 환경에서 GitOps 워크플로를 구현할 수 있습니다. SDK와 Argo CD 또는 Flux와 같은 전용 GitOps 도구의 주요 차이점은 고급 배포 전략과 다중 클라우드 지원을 통해 보다 포괄적인 지속적 제공 플랫폼을 제공한다는 것입니다.

회사의 강점은 다양한 클라우드 제공업체에서 복잡한 배포 시나리오를 처리할 수 있는 역량과 고급 배포 전략에 대한 지원에 있습니다. 적절히 구성되면 GitOps 원칙을 효과적으로 구현할 수 있습니다. 따라서 다양하고 복잡한 환경에서 GitOps 관행을 채택하려는 조직을 위한 강력한 도구입니다.

자세한 내용은 JD 설명서를 참조하세요.

아키텍처

다음 다이어그램은 JD 및 Jenkins X를 사용하는 GitOps 기반 CD 워크플로를 보여줍니다. 자세한 내용은 JD 설명서를 참조하세요.



여기서 각 항목은 다음과 같습니다.

- 1단계: 코드 커밋. 개발자는 애플리케이션 코드 변경 사항을 Git 리포지토리에 커밋합니다. 이러한 변경 사항에는 애플리케이션 자체, Dockerfiles 또는 Kubernetes 매니페스트에 대한 업데이트가 포함될 수 있습니다.
- 2단계: Jenkins 빌드 및 이미지 생성. Jenkins는 웹후크 또는 폴링을 통해 Git 리포지토리에 의해 자동으로 트리거됩니다. Jenkins는 애플리케이션을 빌드하고, Docker 이미지를 생성하고, 빌드된 이미지를 구성된 Docker 레지스트리(예: Amazon ECR 또는 Docker Hub)에 푸시합니다.
- 3단계: 이미지 모니터링 및 파이프라인 트리거. 새 이미지가 있는지 Docker 레지스트리를 지속적으로 모니터링합니다. 새 이미지 버전이 감지되면 자동으로 파이프라인을 트리거하여 배포 프로세스를 시작합니다.
- 4단계: 대상 네임스페이스에 배포. 새 Docker 이미지를 Amazon EKS에 배포합니다. 파이프라인 구성에 따라 이미지가 클러스터의 대상 네임스페이스에 배포됩니다. 블루/그린 또는 카나리 배포와 같은 정의된 배포 전략을 준수하면서 최신 애플리케이션 버전이 배포되도록 보장합니다.

랜처 플릿

Rancher 플릿은 여러 Kubernetes 클러스터를 관리하도록 특별히 설계된 GitOps-at-scale 솔루션입니다. 확장성 및 다중 클러스터 관리에 중점을 두면서 GitOps 원칙을 긴밀히 준수합니다.

GitOps 지원

영역	도구 기능
Git을 신뢰할 수 있는 단일 소스로 사용	플릿은 Git 리포지토리를 여러 클러스터에서 원하는 애플리케이션 및 리소스 상태를 정의하는 신뢰할 수 있는 소스로 사용합니다. Kubernetes 매니페스트, 차트 Helm, 사용자 지정 리소스를 포함한 모든 구성은 Git에 저장됩니다.
선언적 구성	플릿은 애플리케이션 및 리소스의 원하는 상태에 대한 선언적 설명과 함께 작동합니다. 원시 Kubernetes YAML, 차트 Helm, Kustomize 파일 또는 플릿별 사용자 지정 리소스일 수 있습니다.
자동 동기화	플릿은 Git 리포지토리의 변경 사항을 지속적으로 모니터링합니다. Git 상태와 클러스터 상태 간의 차이를 감지하면 대상 클러스터에 변경 사항을 자동으로 적용합니다.
다중 클러스터 관리	플릿은 여러 Kubernetes 클러스터에서 배포를 관리하도록 특별히 설계되었습니다. 단일 컨트롤 플레인에서 수천 개의 클러스터를 처리할 수 있습니다.
Kubernetes 네이티브 아키텍처	플릿은 Kubernetes 사용자 지정 리소스 및 컨트롤러 세트로 빌드됩니다. GitOps 작업에 Kubernetes의 확장 메커니즘을 사용합니다.
지속적 조정	플릿은 클러스터의 실제 상태를 Git에 정의된 원하는 상태와 지속적으로 비교합니다. 이러한 상태 간에 감지된 모든 드리프트를 자동으로 수정합니다.
클러스터 그룹화 및 대상 지정	플릿을 사용하면 클러스터를 그룹화하고 특정 그룹 또는 개별 클러스터에 대한 배포를 대상으로 지정할 수 있습니다. 다양한 환경 및 클러스

영역	도구 기능
	터 유형에서 일관된 애플리케이션 배포를 지원합니다.
계층형 구성	플릿은 환경별 오버레이로 기본 구성을 제공하는 계층화된 구성을 지원합니다. 이는 여러 환경을 효율적으로 관리하는 GitOps 관행에 부합합니다.
Helm 통합	플릿은 차트 Helm에 대한 기본 지원을 제공하고 복잡한 애플리케이션을 쉽게 관리할 수 있습니다. GitOps 워크플로를 통해 Helm 릴리스의 버전을 지정하고 관리할 수 있습니다.
사용자 지정 리소스 정의(CRDs)	플릿은 GitRepo 및 번들과 같은 사용자 지정 리소스를 사용하여 배포를 정의합니다. 이러한 CRDs GitOps 워크플로를 정의하는 Kubernetes 네이티브 방법을 제공합니다.
보안 및 RBAC	플릿은 액세스 제어를 위해 Kubernetes RBAC와 통합됩니다. 민감한 정보 및 자격 증명의 보안 관리를 지원합니다.
관찰성	플릿은 클러스터 및 애플리케이션의 동기화 상태에 대한 상태 정보를 제공합니다. 클러스터 플릿의 GitOps 프로세스에 대한 인사이트를 제공합니다.
확장성	플릿은 수천 개의 클러스터를 효율적으로 관리하도록 확장되도록 설계되었습니다. 엔터프라이즈 환경에서 대규모 GitOps 작업을 지원합니다.
종속성 관리	서로 다른 리소스와 애플리케이션 간의 종속성을 정의할 수 있습니다. 플릿은 복잡한 배포에서 올바른 작업 순서를 따르도록 합니다.

영역	도구 기능
사용자 지정 및 확장성	플릿은 배포의 고급 사용자 지정을 위한 사용자 지정 스크립트 및 수명 주기 후크를 지원합니다. 이를 통해 기존 도구 및 워크플로와 통합할 수 있습니다.
오프라인 및 에어 갭 지원	플릿은 인터넷 연결이 제한되거나 없는 환경에서 작동할 수 있습니다. 높은 보안 또는 규제 환경에서 GitOps 워크플로를 지원합니다.
점진적 롤아웃	플릿은 클러스터 간 단계적 롤아웃을 지원하므로 제어되고 점진적인 배포 전략이 가능합니다.
통합 관리 인터페이스	플릿은 모든 클러스터에서 GitOps 워크플로를 관리하기 위한 단일 인터페이스를 제공합니다. 복잡한 다중 클러스터 환경에서 작업을 간소화합니다.
다른 Rancher 도구와의 통합	플릿은 다른 Rancher 도구와 통합되어 포괄적인 Kubernetes 관리 솔루션을 제공합니다.
감사 추적 및 규정 준수	플릿은 모든 변경 사항 및 배포에 대한 명확한 감사 추적을 유지합니다. 버전 관리형 Git 기반 작업을 통해 규정 준수 요구 사항을 충족하는 데 도움이 됩니다.

Rancher Fleet은 확장성 및 다중 클러스터 관리에 중점을 두고 이러한 GitOps 원칙을 구현합니다. 설계는 다양한 환경, 데이터 센터 또는 클라우드 공급자에서 많은 수의 Kubernetes 클러스터를 관리하는 조직에 특히 적합합니다.

플릿의 주요 차별화 요소는 대규모로 GitOps를 처리하는 기능입니다. 이 기능을 사용하면 수많은 클러스터를 관리하는 대기업 또는 관리형 서비스 공급자에게 특히 유용합니다. Argo CD 또는 Flux와 같은 도구는 개별 클러스터 관리에 자주 사용되는 반면, 플릿은 대규모 클러스터 플릿에서 GitOps를 관리하도록 설계되었습니다.

Rancher Fleet은 이러한 GitOps 원칙을 준수함으로써 다양한 대규모 Kubernetes 환경에서 애플리케이션 및 리소스의 일관되고 확장 가능하며 자동화된 관리를 구현하려는 조직을 위한 솔루션을 제공합니다.

자세한 내용은 플릿 설명서를 참조하세요.

아키텍처

아키텍처 및 워크플로 정보는 플릿 리포지토리를 참조하세요.

Codefresh

Codefresh는 특히 Kubernetes 배포를 위해 GitOps 원칙을 지원하는 최신 CI/CD 플랫폼입니다. Codefresh는 포괄적인 CI/CD 기능 세트를 제공하며 GitOps 기능은 주목할 만합니다.

GitOps 지원

영역	도구 기능
Git을 신뢰할 수 있는 단일 소스로 사용	Codefresh는 Git 리포지토리를 애플리케이션 코드, 인프라 정의 및 파이프라인 구성의 신뢰할 수 있는 소스로 사용합니다. 시스템에 대한 모든 변경은 Git을 통해 이루어지므로 전체 기록 및 감사 추적이 보장됩니다.
선언적 구성	Codefresh는 Git에 저장된 YAML 파일을 사용하여 선언적 파이프라인 정의를 지원합니다. Kubernetes 매니페스트, 차트 Helm, CloudFormation 템플릿 및 기타 IaC 파일은 동일한 리포지토리에서 버전을 제어할 수 있습니다.
GitOps 대시보드	Codefresh는 GitOps 워크플로를 시각화하고 관리하기 위한 전용 GitOps 대시보드를 제공합니다. Git과 클러스터 상태 간의 동기화 상태를 명확하게 볼 수 있습니다.
자동 동기화	Codefresh는 Git 리포지토리의 변경 사항을 지속적으로 모니터링합니다. 차이를 감지하면 대

영역	도구 기능
	상 환경에 변경 사항을 적용하기 위해 파이프라인을 자동으로 시작합니다.
Kubernetes 통합	Codefresh는 Kubernetes와의 심층 통합을 제공하여 여러 클러스터에서 GitOps 스타일 배포를 지원합니다. 다양한 Kubernetes 리소스와 사용자 지정 리소스 정의(CRDs 지원합니다.
환경 관리	여러 환경(예: 개발, 스테이징 및 프로덕션)을 코드로 정의하고 관리할 수 있습니다. Codefresh는 GitOps 사례를 사용하여 환경 간 승격을 지원합니다.
Argo CD 통합	Codefresh는 향상된 GitOps 기능을 위해 Argo CD와 통합됩니다. CI 기능과 Argo CD의 CD 강점을 결합하여 완전한 GitOps 솔루션을 제공합니다.
Helm 지원	Codefresh는 Helm 차트를 지원하며 GitOps를 통해 복잡한 애플리케이션을 쉽게 관리할 수 있습니다. Helm 차트 버전 관리 및 홍보도 제공합니다.
점진적 전달	Codefresh는 canary 및 블루/그린 배포와 같은 고급 배포 전략을 지원합니다. GitOps 워크플로를 통해 이러한 전략을 구현하고 관리할 수 있습니다.
롤백 및 버전 관리	Codefresh를 사용하면 배포 후 문제가 감지되면 이전 버전으로 쉽게 롤백할 수 있습니다. 추적성을 위해 배포 버전 관리를 유지합니다.
승인 워크플로	Codefresh는 배포에 대한 수동 및 자동 승인 프로세스를 지원합니다. GitOps 관행에 따라 환경 간에 제어된 프로모션을 활성화합니다.

영역	도구 기능
IaC	Codefresh는 CloudFormation 및 Terraform과 같은 IaC 도구와의 통합을 지원합니다. 애플리케이션 코드와 함께 인프라 정의의 버전 관리를 활성화합니다.
관찰성 및 모니터링	Codefresh는 내장된 모니터링 및 관찰성 기능을 제공합니다. 또한 가시성을 높이기 위해 외부 모니터링 도구와의 통합을 제공합니다.
보안 스캔	Codefresh에는 GitOps 워크플로에 통합할 수 있는 보안 스캔 기능이 포함되어 있습니다. 보안 검사는 자동 배포 프로세스의 일부입니다.
감사 추적	Codefresh는 모든 작업 및 변경 사항에 대한 포괄적인 감사 로그를 유지합니다. GitOps의 추적성 및 규정 준수 측면을 지원합니다.
RBAC 및 액세스 제어	Codefresh는 세분화된 권한 관리를 위해 역할 기반 액세스 제어(RBAC)를 구현합니다. 이를 통해 팀과 환경 전반에서 GitOps 작업을 보호할 수 있습니다.
GitOps 자동화	Codefresh는 풀 요청(PR) 생성 및 병합을 포함하여 GitOps 워크플로의 다양한 측면을 자동화하는 기능을 제공합니다.
다중 클라우드 및 하이브리드 배포	Codefresh는 여러 클라우드 공급자 및 온프레미스 환경에서 GitOps 워크플로를 지원합니다.
템플릿 지정 및 파라미터화	Codefresh는 파이프라인 및 배포 구성에서 템플릿을 지원합니다. 이렇게 하면 재사용 가능하고 파라미터화된 GitOps 워크플로가 활성화됩니다.
통합 이미지 관리	Codefresh는 기본 제공 컨테이너 이미지 관리 기능을 제공합니다. 이미지 빌드 및 배포를 GitOps 워크플로에 통합합니다.

영역	도구 기능
보안 암호 관리를 위한 GitOps	Codefresh는 GitOps 워크플로 내에서 보안 암호를 관리하는 안전한 방법을 제공합니다. 외부 보안 암호 관리 솔루션과 통합됩니다.
공동 작업 기능	Codefresh는 GitOps 프로세스 내에서 팀 협업을 위한 기능을 제공합니다. 이러한 기능에는 설명, 알림 및 공유 대시보드가 포함됩니다.

GitOps에 대한 Codefresh 접근 방식은 CI/CD 기능을 GitOps 관행과 통합한 것으로 유명합니다. GitOps 원칙을 준수하면서 전체 소프트웨어 제공 수명 주기를 포괄하는 포괄적인 플랫폼을 제공하는 것을 목표로 합니다.

GitOps 영역에서 Codefresh의 주요 차별화 요소는 CI 기능을 CD 및 GitOps 기능과 결합하는 통합 플랫폼 접근 방식입니다. 따라서 GitOps 관행을 구현하면서 복잡한 CI/CD 시나리오를 처리할 수 있는 all-in-one 솔루션을 원하는 팀에 특히 적합합니다.

Codefresh는 특히 Kubernetes 및 클라우드 네이티브 기술로 작업할 때 더 광범위한 CI/CD 컨텍스트 내에서 GitOps 방법론을 채택하려는 조직을 위한 플랫폼을 제공합니다.

자세한 내용은 Codefresh 설명서를 참조하세요.

Pulumi

Pulumi는 GitOps 전용으로 설계되지 않은 IaC 플랫폼입니다. 그러나 특히 클라우드 인프라 및 Kubernetes 배포의 경우 GitOps 원칙을 구현하는 데 효과적으로 사용할 수 있습니다.

GitOps 지원

영역	도구 기능
IaC	Pulumi를 사용하면 Python, TypeScript, Go와 같은 범용 프로그래밍 언어를 사용하여 인프라를 정의할 수 있습니다. 이 코드 기반 접근 방식은 버전이 지정된 선언적 구성에 대한 GitOps 강조와 일치합니다.

영역	도구 기능
Git을 신뢰할 수 있는 단일 소스로 사용	Pulumi의 인프라 코드는 Git 리포지토리에 저장하고 버전을 제어할 수 있습니다. 이렇게 하면 Git이 인프라 정의에 대한 신뢰할 수 있는 단일 소스 역할을 할 수 있습니다.
선언적 원하는 상태	Pulumi는 프로그래밍 언어를 사용하지만 여전히 원하는 인프라 상태를 선언적으로 설명합니다. 코드는 인프라를 생성하기 위한 step-by-step 프로세스가 아니라 인프라의 모습을 정의합니다.
자동 동기화	Pulumi를 CI/CD 파이프라인과 통합하여 Git에서 코드가 업데이트될 때 변경 사항을 자동으로 적용할 수 있습니다. 이를 통해 주요 GitOps 원칙인 인프라 변경 사항을 지속적으로 배포할 수 있습니다.
다중 클라우드 및 Kubernetes 지원	Pulumi는 다양한 클라우드 공급자와 Kubernetes를 지원하므로 다양한 환경에서 GitOps 관행을 따를 수 있습니다. 이 도구를 사용하면 다양한 플랫폼에서 리소스를 일관되게 관리할 수 있습니다.
상태 관리	Pulumi는 원격으로 안전하게 저장할 수 있는 인프라 상태를 관리합니다. 이 상태 관리는 GitOps 관행에 매우 중요하며, 정의된 상태와 인프라의 실제 상태 간의 일관성을 보장합니다.
드리프트 감지 및 조정	Pulumi는 원하는 상태(코드)와 실제 인프라 상태 간의 차이를 감지할 수 있습니다. 지속적인 조정을 위한 GitOps 원칙에 따라 이러한 차이를 조정합니다.
코드형 정책	Pulumi CrossGuard를 사용하여 정책을 코드로 정의하고 적용할 수 있습니다. 이를 통해 버전 제어 GitOps 스타일의 규정 준수 및 보안 정책 관리가 가능합니다.

영역	도구 기능
보안 암호 관리	Pulumi는 인프라 코드 내에서 민감한 정보를 관리하는 안전한 방법을 제공합니다. GitOps 보안 사례에 중요한 외부 보안 암호 관리 시스템과의 통합을 지원합니다.
모듈식 및 재사용 가능한 구성 요소	Pulumi는 재사용 가능한 구성 요소 및 모듈 생성을 지원합니다. 이 모듈화는 복잡한 다중 환경 배포를 관리하기 위한 GitOps 관행에 부합합니다.
미리 보기 및 계획	Pulumi는 변경 사항을 적용하기 전에 미리 볼 수 있는 기능을 제공합니다. 이는 인프라에 대한 안전하고 예측 가능한 변경이라는 GitOps 원칙을 지원합니다.
롤백 및 기록	Pulumi는 배포 기록을 유지 관리하고 이전 상태로의 롤백을 지원합니다. 이는 추적성 및 재현성에 대한 GitOps 원칙에 부합합니다.
인프라에 대한 지속적인 제공	인프라 변경을 지속적으로 제공하기 위해 Pulumi를 CI/CD 파이프라인에 통합할 수 있습니다. 인프라 코드의 자동 테스트 및 검증을 지원합니다.
RBAC 및 액세스 제어	Pulumi는 인프라를 변경할 수 있는 사용자를 관리하기 위한 역할 기반 액세스 제어를 제공합니다. 이는 GitOps 보안 및 거버넌스 관행을 지원합니다.
관찰성 및 로깅	Pulumi는 인프라 변경에 대한 로깅 및 모니터링 기능을 제공합니다. 이러한 기능은 GitOps 관행의 관찰성 측면을 지원합니다.
다른 도구와의 통합	Pulumi는 클라우드의 다양한 도구와 통합할 수 있습니다. 이러한 유연성을 통해 포괄적인 GitOps 워크플로가 가능합니다.

영역	도구 기능
환경 관리	Pulumi는 구성이 다른 동일한 코드베이스를 사용하여 여러 환경(개발, 스테이징, 프로덕션)의 관리를 지원합니다. 이는 일관된 다중 환경 관리를 위한 GitOps 관행에 부합합니다.
종속성 관리	Pulumi는 리소스 간의 종속성을 처리하고 올바른 작업 순서를 보장합니다. 이는 상호 종속적인 구성 요소가 포함된 복잡한 GitOps 배포에 매우 중요합니다.
사용자 지정 리소스 공급자	Pulumi를 사용하면 사용자 지정 공급자를 생성하여 API 기반 서비스를 관리할 수 있습니다. 이를 통해 GitOps 관행이 표준 클라우드 오퍼링을 넘어 광범위한 리소스로 확장됩니다.
공동 작업 기능	Pulumi는 공유 상태 및 액세스 제어를 통해 팀 협업을 지원합니다. 이렇게 하면 팀 환경에서 GitOps 워크플로가 용이해집니다.

조직은 이러한 Pulumi 기능을 사용하여 인프라에 대한 GitOps 사례를 구현할 수 있습니다. 특히 세분화된 제어 또는 복잡한 로직이 필요하거나 일관된 단일 프레임워크 내에서 다양한 클라우드 및 온프레미스 리소스 세트를 관리하려는 시나리오에서 더욱 그렇습니다.

GitOps에 대한 Pulumi의 접근 방식은 GitOps 원칙을 준수하면서 인프라 관리에 범용 프로그래밍 언어의 성능과 유연성을 제공하기 때문에 고유합니다. 이는 익숙한 프로그래밍 언어로 작업하고 인프라 관리에 소프트웨어 엔지니어링 모범 사례를 적용하려는 팀에 특히 유리할 수 있습니다.

GitOps에서 Pulumi의 주요 차별화 요소는 표준 프로그래밍 언어를 사용하여 인프라를 정의하는 것입니다. 기존 GitOps 도구는 YAML 또는 도메인별 언어를 사용하는 경우가 많지만, Pulumi를 사용하면 더 복잡한 로직, 더 나은 코드 재사용, 기존 개발 워크플로와의 더 쉬운 통합이 가능합니다.

자세한 내용은 Pulumi 설명서를 참조하세요.

GitOps 도구 비교

다음은 이전 섹션에서 설명한 9가지 GitOps 도구의 비교입니다. 도구를 선택할 때 특정 요구 사항, 기존 인프라, 팀 전문 지식, 원하는 수준의 제어 및 사용자 지정을 고려합니다.

사용 편의성

- Argo CD, Flux 및 Rancher 플릿은 일반적으로 설정하기가 더 쉽습니다.
- Spinnaker와 Jenkins X는 학습 곡선이 더 가파릅니다.
- Weave GitOps는 고급 기능을 더 많이 설정해야 할 수 있습니다.
- GitLab CI/CD 및 Codefresh는 통합된 경험을 제공합니다.

Kubernetes 통합

- Argo CD, Flux 및 Rancher 플릿은 매우 Kubernetes 중심입니다.
- Jenkins X 및 Weave GitOps는 더 광범위한 DevOps 기능을 제공합니다.
- 다른 도구는 Kubernetes를 독점적으로 집중하지 않고 지원합니다.

CI/CD 기능

- Jenkins X, GitLab CI/CD 및 Codefresh는 완전한 CI/CD 솔루션을 제공합니다.
- Argo CD, Flux 및 Weave GitOps는 워크플로의 CD 측면에 더 중점을 두며, 종종 별도의 CI 도구와의 통합이 필요합니다.

GitOps 내결함성

- Argo CD 및 Flux는 특히 GitOps에 초점을 맞춘 도구입니다.
- 다른 도구는 GitOps 원칙을 다양한 수준으로 통합합니다.

다중 클라우드 지원

- 멀티클라우드 시나리오에서 뛰어난 성능을 발휘합니다.

- 다른 도구는 클라우드에서 작동할 수 있지만 추가 설정이 필요할 수 있습니다.

다중 클러스터 지원

- 모든 도구는 다중 클러스터 배포를 지원합니다.
- Argo CD 및 Weave GitOps에는 고급 다중 클러스터 관리 기능이 있습니다.

통합

- Flux에는 강력한 Cloud Native Computing Foundation(CNCF) 지원 기능이 있습니다.
- Argo CD에는 크고 활동적인 커뮤니티가 있습니다.
- Argo CD와 Flux에는 강력한 Kubernetes 통합이 있습니다.
- Jenkins X는 더 광범위한 Jenkins 시스템을 사용합니다.
- Weave GitOps는 최신 버전이지만 강력한 상용 지원으로 성장하고 있습니다.
- GitLab CI/CD는 GitLab과 긴밀하게 통합됩니다.
- Rancher 플릿은 Rancher 시스템 내에서 잘 작동합니다.

커뮤니티 및 지원

- Flux에는 강력한 CNCF 지원 기능이 있습니다.
- Argo CD, GitLab 및 Chef에는 대규모 커뮤니티가 있습니다.
- 상용 지원은 대부분의 도구에서 사용할 수 있습니다.

엔터프라이즈 기능

- Weave GitOps 및 Jenkins X는 기본적으로 더 많은 엔터프라이즈 중심 기능을 제공합니다.
- Argo CD 및 Flux에는 엔터프라이즈 제품이 있거나 엔터프라이즈용으로 확장할 수 있습니다.

유연성 및 확장성

- Flux는 모듈식이며 확장 가능합니다.
- Argo CD는 좋은 사용자 지정 옵션을 제공합니다.

- Jenkins X는 매우 확장 가능하지만 더 많은 노력이 필요할 수 있습니다.
- Weave GitOps는 확장성의 필요성을 줄이면서 완전한 솔루션을 제공하는 것을 목표로 합니다.

확장성

- 기업 확장성으로 유명 GitLab 합니다.
- Argo CD 및 Flux는 대규모 Kubernetes 배포를 잘 처리합니다.

인프라 관리

- Pulumi는 인프라 관리에 중점을 둡니다.
- Weave GitOps 및 Flux는 우수한 IaC 기능을 제공합니다.

프로그래밍 모델 및 언어 지원

- Pulumi에서는 Python, Go, TypeScript, C#, Java와 같은 범용 프로그래밍 언어를 사용하여 인프라를 정의할 수 있습니다. Pulumi는 표준 언어를 사용하여 인프라 코드를 익숙한 개발 워크플로, 테스트 관행 및 복잡한 로직과 통합할 수 있습니다.
- Terraform은 HashiCorp 구성 언어(HCL)를 사용합니다.
- CloudFormation 는 JSON 및 YAML 템플릿을 사용합니다.
- Argo CD, Flux, Rancher 플릿, Weave GitOps, Siebel 및 GitLab CI/CD는 주로 YAML 또는 선언적 구성 파일을 관리합니다.
- Jenkins X는 YAML 및 스크립팅 기반 파이프라인을 관리하지만 기본적으로 IaC용 범용 프로그래밍을 제공하지 않습니다.

Argo CD 및 Flux 사용 사례

이 섹션에서는 순수 GitOps 기능을 제공하는 Argo CD와 Flux라는 두 가지 도구에 중점을 둡니다. 이 컨텍스트에서 순수 GitOps는 Git 리포지토리가 원하는 애플리케이션 및 인프라 상태에 대한 단일 신뢰할 수 있는 소스 역할을 하는 모델을 말합니다. 모든 변경 사항은 Git 커밋을 통해 수행되며 시스템은 리포지토리에 정의된 상태와 일치하도록 라이브 환경을 자동으로 동기화합니다. Git 작업 외부에서는 수동 개입이 필요하지 않습니다.

일반적인 고려 사항

- 시각적 관리 및 애플리케이션 중심 워크플로가 중요한 환경에서는 Argo CD를 사용하는 것이 좋습니다.
- 경량 솔루션, 강력한 다중 테넌시 또는 광범위한 Cloud Native Computing Foundations(CNCF) 네트워크와의 심층 통합이 필요한 경우 Flux를 선택할 수 있습니다.
- Argo CD는 직관적인 UI로 인해 기존 CI/CD에서 GitOps로 전환하는 팀에 자주 호소합니다.
- Flux는 CLI 기반 워크플로 및 IaC 관행이 이미 설정된 클라우드 네이티브 환경에서 선호되는 경우가 많습니다.

궁극적으로 Argo CD와 Flux 간의 선택은 특정 조직의 요구 사항, 기존 도구 및 팀 기본 설정에 따라 달라지는 경우가 많습니다. 두 도구 모두 대부분의 GitOps 시나리오를 처리할 수 있으므로 특정 사용 사례 및 요구 사항에 따라 평가하는 것이 좋습니다.

Argo CD 사용 사례

시각적 관리:

- 배포를 관리하고 애플리케이션 상태를 시각화하기 위해 사용자 친화적인 UI가 필요한 경우.
- 모니터링 및 문제 해결을 위해 그래픽 인터페이스를 선호하는 팀의 경우.

애플리케이션 중심 접근 방식:

- 개별 리소스를 관리하는 대신 애플리케이션 수준에서 배포를 관리하려는 경우.
- 애플리케이션 개념을 중심으로 배포를 구성하는 조직의 경우.

다중 클러스터 관리:

- 여러 클러스터에서 배포를 관리하는 것은 기본 요구 사항입니다.
- 클러스터가 많은 복잡한 분산 환경의 경우.

롤백 및 동기화 웨이브:

- 동기화 웨이브 및 수동 개입을 포함하여 배포 프로세스를 세밀하게 제어해야 하는 경우.
- 복잡한 롤백 전략이 필요한 시나리오의 경우.

기존 도구와의 통합:

- Argo 워크플로 및 Argo 이벤트와 같은 Argo 프로젝트에서 다른 도구를 이미 사용하고 있는 경우.

엔터프라이즈 환경:

- 기본적으로 강력한 RBAC 및 Single Sign-On 통합이 필요한 대기업의 경우.

Flux 사용 사례

경량 배포:

- 더 가볍고 리소스 집약적이지 않은 GitOps 솔루션이 필요한 경우.
- 리소스가 제한될 수 있는 엣지 컴퓨팅 또는 IoT 시나리오의 경우.

자동 이미지 업데이트:

- 새 컨테이너 이미지의 자동 감지 및 배포가 주요 요구 사항인 경우.
- 빈번한 이미지 업데이트로 지속적인 배포에 집중하는 팀의 경우.

다중 테넌시:

- 특히 공유 클러스터 환경에서 강력한 다중 테넌시 지원이 필요한 경우.
- 팀 또는 프로젝트 간에 엄격한 분리가 있는 서비스 공급자 또는 대규모 조직의 경우.

IaC:

- 동일한 GitOps 워크플로를 통해 애플리케이션과 인프라를 모두 관리하는 것이 중요합니다.
- IaC 패러다임에 많은 투자를 하는 팀의 경우.

Helm 통합:

- 차트 Helm을 광범위하게 사용하는 것이 배포 전략의 일부인 경우.
- 복잡한 Helm 기반 배포가 있는 환경의 경우.

CNCF 프로젝트 통합:

- 다른 CNCF 프로젝트와의 긴밀한 통합이 중요한 경우.
- CNCF 기술 및 원칙에 부합하는 조직의 경우.

모듈식 아키텍처:

- GitOps 툴킷의 특정 구성 요소만 사용할 수 있는 유연성이 필요한 경우.
- 모듈식 구성 요소를 사용하여 사용자 지정 GitOps 워크플로를 구축하려는 팀의 경우.

점진적 전달:

- canary 릴리스 또는 A/B 테스트와 같은 고급 배포 전략이 핵심 요구 사항인 경우.

기능 비교

영역	Argo CD	Flux
핵심 GitOps 원칙 지원	✓예	✓예
아키텍처	Kubernetes GitOps 워크플로를 구현End-to-end 애플리케이션	GitOps용 Kubernetes CRDs 및 컨트롤러 제공
설정	간편함	복합
Helm 지원	✓예	✓예

영역	Argo CD	Flux
Kustomize 지원	✅예	✅예
통합 GUI	CLI 및 완전한 기능을 갖춘 웹 UI	CLI 및 선택적 경량 웹 인터페이스
RBAC 지원	세분화된 제어	Kubernetes 네이티브 RBAC
다중 테넌시 및 다중 클러스터 지원	다중 클러스터에 대한 뛰어난 지원	다중 테넌시에 대한 뛰어난 지원
Single Sign-On 인증	✅예	✅예
동기화 자동화	창 동기화 기능	조정 간격 설정 기능
부분 동기화	✅예	❌아니요
조정 프로세스	수동 및 자동 동기화를 지원합니다. 다양한 전략을 사용할 수 있습니다.	수동 및 자동 동기화를 지원합니다.
확장성	사용자 지정 lugins를 지원합니다. 제한된 사용자 지정 옵션.	사용자 지정 컨트롤러를 지원합니다. 우수한 확장성 및 타사 통합.
Cummunity 지원	크고 활동적인 커뮤니티.	작지만 성장하는 커뮤니티.
확장성	확장성은 좋지만 웹 UI의 데이터 가져오기 속도에 따라 제한됩니다. 커뮤니티 분석은 수만 개의 애플리케이션에 대한 지원을 제안합니다.	수평 및 수직 확장성, 최대 수만 개의 애플리케이션에 대한 명확한 가이드입니다.

GitOps 도구 선택 모범 사례

이 섹션에서는 EKS 클러스터용 GitOps 도구를 선택하기 위한 고려 사항, 팁 및 모범 사례를 제공합니다. 올바른 선택은 특정 컨텍스트, 요구 사항 및 장기 전략에 따라 달라집니다. 최종 결정을 내리기 전에 상위 선택 항목으로 개념 증명을 수행하는 것이 유용한 경우가 많습니다.

조직의 요구 사항과 기능을 평가합니다.

- 팀의 현재 기술과 새로운 도구를 배우려는 의지를 고려합니다.
- Amazon EKS 환경의 복잡성을 평가합니다. (예: 단일 클러스터를 사용하고 있습니까, 아니면 여러 클러스터를 사용하고 있습니까?)
- 규정 준수, 보안 및 확장성에 대한 특정 요구 사항을 결정합니다.

> **모범 사례**
>
> 필수 기능과 유용하지만 필수는 아닌 기능을 간략하게 설명하는 세부 요구 사항 문서를 생성합니다.

도구 성숙도 및 채택을 평가합니다.

- 업계에서 잠재적 GitOps 도구의 성숙도와 채택률을 조사합니다.
- Amazon EKS 환경에서 트랙 기록이 검증된 도구를 찾습니다.

> **모범 사례**
>
> 널리 채택되어 클라우드 네이티브 컴퓨팅 파운데이션(CNCF) 네트워크에 강력한 입지를 갖춘 도구의 우선순위를 정합니다.

기존 도구 체인과의 통합을 고려합니다.

- GitOps 도구가 현재 CI/CD 파이프라인, 모니터링 솔루션 및 기타 운영 도구와 얼마나 잘 통합되는지 평가합니다.
- IAM, Amazon ECR 및 CloudWatch와 AWS 서비스 같은 와의 네이티브 통합을 찾습니다.

> **모범 사례**
>
> 최종 결정을 내리기 전에 통합 기능을 테스트하는 개념 증명을 생성합니다.

보안 기능 평가:

- 강력한 역할 기반 액세스 제어(RBAC) 기능이 있고 IAM과 잘 통합되는 도구의 우선순위를 정합니다.
- 보안 암호 관리 및 정책 적용을 지원하는 기능을 찾습니다.

> **모범 사례**
>
> 코드형 정책 및 자동화된 규정 준수 검사를 포함하여 GitOps 기반 보안 관행을 지원하는 도구를 선택합니다.

확장성 및 성능 평가:

- 도구가 많은 수의 애플리케이션 및 클러스터에서 어떻게 작동하는지 고려합니다.
- 클러스터 성능 및 리소스 소비에 미치는 영향을 평가합니다.

> **모범 사례**
>
> 프로덕션 환경과 유사한 워크로드로 성능 테스트를 수행하여 도구가 규모를 처리할 수 있는지 확인합니다.

다중 클러스터 및 다중 환경 지원을 고려합니다.

- 여러 EKS 클러스터가 있거나 있을 계획인 경우 강력한 다중 클러스터 관리 기능이 있는 도구의 우선순위를 지정합니다.
- 다양한 환경(예: 개발, 스테이징 및 프로덕션)에서 일관된 배포를 지원하는 기능을 찾습니다.

> **모범 사례**
>
> 환경별 구성을 유지하면서 여러 클러스터의 중앙 집중식 관리를 허용하는 도구를 선택합니다.

관찰성 및 모니터링 기능 평가:

- 배포 상태 및 클러스터 상태에 대한 명확한 가시성을 제공하는 도구를 찾습니다.
- 도구가 기존 모니터링 및 로깅 솔루션과 얼마나 잘 통합되는지 고려합니다.

> 모범 사례
>
> 사전 문제 감지를 위해 사용자 지정 가능한 대시보드 및 알림 메커니즘을 제공하는 도구의 우선순위를 지정합니다.

학습 곡선 및 설명서를 평가합니다.

- 도구 설명서의 품질과 포괄성을 평가합니다.
- 훈련 리소스 및 커뮤니티 지원의 가용성을 고려합니다.

> 모범 사례
>
> 잘 유지 관리된 설명서, 활성 커뮤니티 포럼, 공식 훈련 프로그램 또는 인증서가 있는 도구를 선택합니다.

비용 및 리소스 사용률을 고려합니다.

- 도구 채택의 직접 비용(예: 라이선스 및 지원)과 간접 비용(예: 운영 오버헤드 및 교육 비용)을 모두 평가합니다.
- 컴퓨팅 및 스토리지 리소스 소비 측면에서 도구의 효율성을 평가합니다.

> 모범 사례
>
> 단기 및 장기 비용을 모두 포함하는 총 소유 비용(TCO) 분석을 수행합니다.

유연성 및 사용자 지정 옵션 평가:

- 특정 요구 사항에 맞게 워크플로를 사용자 지정할 수 있는 도구를 찾습니다.
- 플러그인 또는 APIs를 통해 도구의 확장성을 고려합니다.

> **모범 사례**
>
> 기본 기능과 고유한 요구 사항에 맞게 사용자 지정할 수 있는 기능의 균형을 맞추는 도구를 선택합니다.

지속적 제공 및 점진적 배포 기능 평가:

- canary 릴리스 및 블루/그린 배포와 같은 고급 배포 전략을 지원하는 도구를 찾습니다.
- 이러한 전략을 쉽게 구현하고 관리할 수 있는지 평가합니다.

> **모범 사례**
>
> 점진적 전송 패턴에 대한 기본 지원을 제공하는 도구의 우선순위를 지정하여 배포의 위험을 최소화합니다.

공급업체 잠금 및 이동성을 고려합니다.

- 특정 클라우드 공급자 또는 기술에 대한 도구의 종속성을 평가합니다.
- 필요한 경우 향후 다른 도구로 쉽게 마이그레이션할 수 있습니다.

> **모범 사례**
>
> 개방형 표준을 사용하고 GitOps 구성에 대한 내보내기 기능을 제공하는 도구를 선호합니다.

커뮤니티 지원 및 확장 평가:

- 사용자 커뮤니티의 크기와 활동을 살펴봅니다.
- 타사 통합 및 플러그인의 가용성을 평가합니다.

> **모범 사례**
>
> 커뮤니티 포럼 또는 사용자 그룹에 가입하여 결정을 내리기 전에 다른 사용자로부터 직접 경험을 얻으세요.

규정 준수 및 감사 요구 사항을 고려합니다.

- 도구가 감사 추적 및 보고를 포함하여 규정 준수 요구 사항을 얼마나 잘 지원하는지 평가합니다.
- 규정 준수를 유지하고 입증하는 데 도움이 되는 기능을 찾습니다.

> **모범 사례**
>
> 포괄적인 감사 로그를 제공하고 규정 준수 보고서 생성을 지원하는 도구를 선택합니다.

롤백 및 재해 복구 기능 평가:

- 롤백 메커니즘의 용이성과 신뢰성을 평가합니다.
- 도구가 재해 복구 시나리오를 지원하는 방법을 고려합니다.

> **모범 사례**
>
> 평가의 일환으로 롤백 및 복구 프로세스를 철저히 테스트합니다.

FAQ

Q: Amazon EKS에서 가장 인기 있는 GitOps 도구는 무엇입니까?

A: Amazon EKS에서 가장 널리 사용되는 GitOps 도구에는 Argo CD, Flux, Jenkins X 및 GitLab CI/CD가 포함됩니다. 각 도구에는 장점이 있지만 Argo CD 및 Flux는 Kubernetes 네이티브 접근 방식과 강력한 커뮤니티 지원으로 특히 잘 알려져 있습니다.

Q: GitOps는 EKS 클러스터 관리를 어떻게 개선하나요?

A: GitOps는 인프라에 대한 버전 제어, 자동화된 배포, 선언적 구성을 통한 보안 개선, 더 쉬운 롤백, 더 나은 감사 기능을 제공하여 EKS 클러스터 관리를 개선합니다. 또한 협업을 개선하고 배포 시 인적 오류를 줄입니다.

Q: Amazon EKS용 GitOps 도구에서 어떤 주요 기능을 찾아야 하나요?

A: 확인해야 할 주요 기능에는 원활한 Amazon EKS 통합, 강력한 RBAC, 다중 클러스터 지원, 관찰성 기능, 점진적 전송 전략 지원, 확장성, IAM 및 Amazon ECR과 AWS 서비스 같은 와의 통합이 포함됩니다.

Q: Amazon EKS에서 GitOps를 구현할 때 보안을 보장하려면 어떻게 해야 하나요?

A: 보안을 보장하려면 IAM과의 강력한 RBAC 통합, 보안 암호 관리, 암호화된 Git 리포지토리 지원, 보안 정책을 코드로 구현할 수 있는 기능을 갖춘 도구를 선택합니다. 또한 도구가 포괄적인 감사 로그를 제공하는지 확인합니다.

Q: GitOps 도구가 다중 클러스터 Amazon EKS 환경을 처리할 수 있습니까?

A: 예, Argo CD 및 Flux와 같은 GitOps 도구에는 강력한 다중 클러스터 관리 기능이 있습니다. 이를 통해 단일 컨트롤 플레인에서 여러 EKS 클러스터를 관리할 수 있으므로 환경 간 일관성을 보장할 수 있습니다.

Q: GitOps 도구는 기존 CI/CD 파이프라인과 어떻게 통합되나요?

A: GitOps 도구는 일반적으로 파이프라인의 배포 단계 역할을 하여 기존 CI/CD 파이프라인과 통합됩니다. 변경 사항이 Git 리포지토리로 푸시될 때 CI 도구에 의해 트리거될 수 있으며 EKS 클러스터로 배포 프로세스를 자동화합니다.

Q: Amazon EKS에서 GitOps를 구현할 때 발생하는 문제는 무엇입니까?

A: 일반적인 과제로는 보안 암호의 안전한 관리, 적절한 액세스 제어 보장, 상태 저장 애플리케이션 처리, Git과 클러스터 상태 간의 드리프트 관리, GitOps 모델에 대한 팀 워크플로 조정 등이 있습니다.

Q: GitOps 도구는 Amazon EKS에서 롤백을 어떻게 처리하나요?

A: GitOps 도구는 일반적으로 Git 리포지토리의 이전 커밋으로 되돌려 롤백을 처리합니다. 이렇게 하면 이전에 알려진 정상 상태의 배포가 자동으로 트리거되어 빠르고 안정적인 롤백이 발생합니다.

Q: GitOps 도구가 Amazon EKS 추가 기능 및 기타 AWS 리소스를 관리할 수 있나요?

A: 많은 GitOps 도구가 Amazon EKS 추가 기능과 일부 AWS 리소스를 관리할 수 있습니다. 특히 Terraform 또는와 같은 IaC 도구와 결합된 경우 더욱 그렇습니다 CloudFormation. 그러나이 기능의 범위는 다를 수 있습니다. 각 도구에 대한 자세한 내용은 GitOps 도구 섹션을 참조하세요.

Q: GitOps 도구는 Amazon EKS의 규정 준수 요구 사항을 어떻게 지원하나요?

A: GitOps 도구는 모든 변경 사항에 대한 명확한 감사 추적을 제공하고, 승인 프로세스를 적용하고, 자동화된 규정 준수 검사를 위한 코드로 정책을 구현하고, 자세한 로깅 및 보고 기능을 제공하여 규정 준수를 지원합니다.

Q: Amazon EKS에서 GitOps를 구현하기 위한 학습 곡선은 무엇인가요?

A: 학습 곡선은 도구 및 팀의 기존 지식에 따라 달라질 수 있습니다. 일반적으로 Git, Kubernetes 및 Amazon EKS에 익숙한 팀은 다른 팀보다 더 빠르게 적응합니다. 가장 인기 있는 도구는 채택을 용이하게 하기 위해 광범위한 설명서 및 교육 리소스를 제공합니다.

Q: GitOps 도구는 Amazon EKS에서 보안 암호 관리를 어떻게 처리하나요?

A: GitOps 도구는 일반적으로 AWS Secrets Manager 또는 HashiCorp Vault와 같은 외부 보안 암호 관리 솔루션과 통합됩니다. 일부 도구는 Git 리포지토리에 저장된 보안 암호에 대한 기본 제공 암호화도 제공합니다.

Q: GitOps 도구가 Amazon EKS의 상태 비저장 애플리케이션과 상태 저장 애플리케이션 모두에서 작동할 수 있나요?

A: 예, GitOps 도구는 상태 비저장 애플리케이션과 상태 저장 애플리케이션 모두에서 작동할 수 있습니다. 그러나 상태 저장 애플리케이션을 관리하려면 영구 볼륨 처리 및 업데이트 중 데이터 일관성 보장과 같은 추가 고려 사항이 필요한 경우가 많습니다.

Q: GitOps 도구는 Amazon EKS에서 카나리아 또는 블루/그린 배포를 어떻게 지원하나요?

A: 많은 GitOps 도구가 고급 배포 전략에 대한 기본 지원을 제공합니다. 새 버전의 점진적인 롤아웃을 관리하고, 문제를 모니터링하고, 문제가 감지되면 자동으로 롤백할 수 있습니다. 이러한 모든 작업은 Git 리포지토리에서 코드로 정의됩니다.

Q: GitOps 도구를 사용하는 것과 CI/CD 파이프라인**`kubectl apply`**을 사용하는 것의 차이점은 무엇인가요?

A: GitOps 도구는 자동화된 드리프트 감지 및 조정, 풀 기반 배포를 통한 보안 개선, 감사 가능성 향상, 보다 정교한 배포 전략 등 간단한 `kubectl apply` 명령보다 이점이 있습니다. 또한 전체 클러스터 상태를 관리하는 보다 포괄적인 접근 방식을 제공합니다.

리소스

다음 리소스는 EKS 클러스터용 GitOps 도구를 선택할 때 정보에 입각한 결정을 내리는 데 도움이 되는 공식 설명서, 실제 가이드, 사례 연구 및 심층 분석을 제공합니다. 구현 전략, 모범 사례, 다양한 도구 간의 비교, 실제 경험을 포함하여 GitOps의 다양한 측면을 다룹니다.

AWS 리소스:

- Amazon EKS 설명서
- GitOps를 사용하여 Amazon EKS 자동화(AWS 블로그 게시물)
- Weaveworks를 사용한 EKS의 GitOps 소개(AWS 워크숍)
- Flux 랩(Amazon EKS 워크숍)
- Argo CD 랩(Amazon EKS 워크숍)

GitOps 및 도구 설명서:

- 지속적 배포 및 점진적 보안을 위한 GitOps 모범 사례(온디맨드 DevOps.com 웨비나)
- Kubernetes 설명서
- Argo CD 설명서
- Flux 설명서
- Weave GitOps 설명서
- Jenkins X 설명서
- GitLab CI/CD 설명서
- JD 설명서
- Rancher 플릿 설명서
- Codefresh 설명서

문서 기록

아래 표에 이 가이드의 주요 변경 사항이 설명되어 있습니다. 향후 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
최초 게시	—	2025년 4월 30일

AWS 권장 가이드 용어집

다음은 AWS 권장 가이드에서 제공하는 전략, 가이드 및 패턴에서 일반적으로 사용되는 용어입니다. 용어집 항목을 제안하려면 용어집 끝에 있는 피드백 제공 링크를 사용하십시오.

숫자

7가지 전략

애플리케이션을 클라우드로 이전하기 위한 7가지 일반적인 마이그레이션 전략 이러한 전략은 Gartner가 2011년에 파악한 5가지 전략을 기반으로 하며 다음으로 구성됩니다.

- 리팩터링/리아키텍트 - 클라우드 네이티브 기능을 최대한 활용하여 애플리케이션을 이동하고 해당 아키텍처를 수정함으로써 민첩성, 성능 및 확장성을 개선합니다. 여기에는 일반적으로 운영 체제와 데이터베이스 이식이 포함됩니다. 예: 온프레미스 Oracle 데이터베이스를 Amazon Aurora PostgreSQL 호환 에디션으로 마이그레이션합니다.
- 리플랫포밍(리프트 앤드 리셰이프) - 애플리케이션을 클라우드로 이동하고 일정 수준의 최적화를 도입하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 AWS 클라우드의 Amazon Relational Database Service(Amazon RDS) for Oracle로 마이그레이션합니다.
- 재구매(드롭 앤드 숍) - 일반적으로 기존 라이선스에서 SaaS 모델로 전환하여 다른 제품으로 전환합니다. 예: 고객 관계 관리(CRM) 시스템을 Salesforce.com으로 마이그레이션합니다.
- 리호스팅(리프트 앤드 시프트) - 애플리케이션을 변경하지 않고 클라우드로 이동하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 AWS 클라우드클라우드의 EC2 인스턴스에 있는 Oracle로 마이그레이션합니다.
- 재배치(하이퍼바이저 수준의 리프트 앤 시프트) - 새 하드웨어를 구매하거나, 애플리케이션을 다시 작성하거나, 기존 운영을 수정하지 않고도 인프라를 클라우드로 이동합니다. 온프레미스 플랫폼에서 동일한 플랫폼의 클라우드 서비스로 서버를 마이그레이션합니다. 예: Microsoft Hyper-V 애플리케이션을 로 마이그레이션합니다 AWS.
- 유지(보관) - 소스 환경에 애플리케이션을 유지합니다. 대규모 리팩터링이 필요하고 해당 작업을 나중으로 연기하려는 애플리케이션과 비즈니스 차원에서 마이그레이션할 이유가 없어 유지하려는 레거시 애플리케이션이 여기에 포함될 수 있습니다.
- 사용 중지 - 소스 환경에서 더 이상 필요하지 않은 애플리케이션을 폐기하거나 제거합니다.

A

ABAC

속성 기반 액세스 제어를 참조하세요.

추상화된 서비스

관리형 서비스를 참조하세요.

ACID

원자성, 일관성, 격리성, 내구성을 참조하세요.

능동-능동 마이그레이션

양방향 복제 도구 또는 이중 쓰기 작업을 사용하여 소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되고, 두 데이터베이스 모두 마이그레이션 중 연결 애플리케이션의 트랜잭션을 처리하는 데이터베이스 마이그레이션 방법입니다. 이 방법은 일회성 전환이 필요한 대신 소규모의 제어된 배치로 마이그레이션을 지원합니다. 더 유연하지만 액티브 패시브 마이그레이션보다 더 많은 작업이 필요합니다.

능동-수동 마이그레이션

소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되지만 소스 데이터베이스만 연결 애플리케이션의 트랜잭션을 처리하고 데이터는 대상 데이터베이스로 복제되는 데이터베이스 마이그레이션 방법입니다. 대상 데이터베이스는 마이그레이션 중 어떤 트랜잭션도 허용하지 않습니다.

집계 함수

행 그룹에서 작동하고 그룹에 대한 단일 반환 값을 계산하는 SQL 함수입니다. 집계 함수의 예로 `SUM` 및 `MAX`가 있습니다.

AI

인공 지능을 참조하세요.

AIOps

인공 지능 운영을 참조하세요.

익명화

데이터세트에서 개인 정보를 영구적으로 삭제하는 프로세스입니다. 익명화는 개인 정보 보호에 도움이 될 수 있습니다. 익명화된 데이터는 더 이상 개인 데이터로 간주되지 않습니다.

안티 패턴

솔루션이 다른 솔루션보다 비생산적이거나 비효율적이거나 덜 효과적이어서 반복되는 문제에 자주 사용되는 솔루션입니다.

애플리케이션 제어

맬웨어로부터 시스템을 보호하기 위해 승인된 애플리케이션만 사용하도록 허용하는 보안 접근 방식입니다.

애플리케이션 포트폴리오

애플리케이션 구축 및 유지 관리 비용과 애플리케이션의 비즈니스 가치를 비롯하여 조직에서 사용하는 각 애플리케이션에 대한 세부 정보 모음입니다. 이 정보는 포트폴리오 탐색 및 분석 프로세스의 핵심이며 마이그레이션, 현대화 및 최적화할 애플리케이션을 식별하고 우선순위를 정하는 데 도움이 됩니다.

인공 지능

컴퓨터 기술을 사용하여 학습, 문제 해결, 패턴 인식 등 일반적으로 인간과 관련된 인지 기능을 수행하는 것을 전문으로 하는 컴퓨터 과학 분야입니다. 자세한 내용은 What is Artificial Intelligence?를 참조하십시오.

인공 지능 운영(AIOps)

기계 학습 기법을 사용하여 운영 문제를 해결하고, 운영 인시던트 및 사용자 개입을 줄이고, 서비스 품질을 높이는 프로세스입니다. AWS 마이그레이션 전략에서 AIOps가 사용되는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

비대칭 암호화

한 쌍의 키, 즉 암호화를 위한 퍼블릭 키와 복호화를 위한 프라이빗 키를 사용하는 암호화 알고리즘입니다. 퍼블릭 키는 복호화에 사용되지 않으므로 공유할 수 있지만 프라이빗 키에 대한 액세스는 엄격히 제한되어야 합니다.

원자성, 일관성, 격리성, 내구성(ACID)

오류, 정전 또는 기타 문제가 발생한 경우에도 데이터베이스의 데이터 유효성과 운영 신뢰성을 보장하는 소프트웨어 속성 세트입니다.

ABAC(속성 기반 액세스 제어)

부서, 직무, 팀 이름 등의 사용자 속성을 기반으로 세분화된 권한을 생성하는 방식입니다. 자세한 내용은 AWS Identity and Access Management (IAM) 설명서의 용 ABAC AWS를 참조하세요.

신뢰할 수 있는 데이터 소스

가장 신뢰할 수 있는 정보 소스로 간주되는 기본 버전의 데이터를 저장하는 위치입니다. 익명화, 편집 또는 가명화와 같은 데이터 처리 또는 수정의 목적으로 신뢰할 수 있는 데이터 소스의 데이터를 다른 위치로 복사할 수 있습니다.

가용 영역

다른 가용 영역의 장애로부터 격리 AWS 리전 되고 동일한 리전의 다른 가용 영역에 저렴하고 지연 시간이 짧은 네트워크 연결을 제공하는 내의 고유한 위치입니다.

AWS 클라우드 채택 프레임워크(AWS CAF)

조직이 클라우드로 성공적으로 전환 AWS 하기 위한 효율적이고 효과적인 계획을 개발하는 데 도움이 되는의 지침 및 모범 사례 프레임워크입니다. AWS CAF는 지침을 비즈니스, 사람, 거버넌스, 플랫폼, 보안 및 운영이라는 6가지 중점 영역으로 구성합니다. 비즈니스, 사람 및 거버넌스 관점은 비즈니스 기술과 프로세스에 초점을 맞추고, 플랫폼, 보안 및 운영 관점은 전문 기술과 프로세스에 중점을 둡니다. 예를 들어, 사람 관점은 인사(HR), 직원 배치 기능 및 인력 관리를 담당하는 이해관계자를 대상으로 합니다. 이러한 관점에서 AWS CAF는 성공적인 클라우드 채택을 위해 조직을 준비하는 데 도움이 되는 인력 개발, 교육 및 커뮤니케이션에 대한 지침을 제공합니다. 자세한 내용은 AWS CAF 웹사이트와 AWS CAF 백서를 참조하세요.

AWS 워크로드 검증 프레임워크(AWS WQF)

데이터베이스 마이그레이션 워크로드를 평가하고, 마이그레이션 전략을 권장하고, 작업 견적을 제공하는 도구입니다. AWS WQF는 AWS Schema Conversion Tool (AWS SCT)에 포함되어 있습니다. 데이터베이스 스키마 및 코드 객체, 애플리케이션 코드, 종속성 및 성능 특성을 분석하고 평가 보고서를 제공합니다.

B

악성 봇

개인 또는 조직을 방해하거나 해를 입히기 위한 봇입니다.

BCP

비즈니스 연속성 계획을 참조하세요.

동작 그래프

리소스 동작과 시간 경과에 따른 상호 작용에 대한 통합된 대화형 뷰입니다. Amazon Detective에서 동작 그래프를 사용하여 실패한 로그온 시도, 의심스러운 API 직접 호출 및 유사한 작업을 검사할 수 있습니다. 자세한 내용은 Detective 설명서의 Data in a behavior graph를 참조하십시오.

빅 엔디안 시스템

가장 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안도 참조하세요.

바이너리 분류

바이너리 결과(가능한 두 클래스 중 하나)를 예측하는 프로세스입니다. 예를 들어, ML 모델이 "이 이메일이 스팸인가요, 스팸이 아닌가요?", '이 제품은 책인가요, 자동차인가요?' 등의 문제를 예측해야 할 수 있습니다.

블룸 필터

요소가 세트의 멤버인지 여부를 테스트하는 데 사용되는 메모리 효율성이 높은 확률론적 데이터 구조입니다.

블루/그린(Blue/Green) 배포

동일하지만 별개의 두 환경을 생성하는 배포 전략입니다. 하나의 환경(파란색)에서 현재 애플리케이션 버전을 실행하고 새 애플리케이션 버전은 다른 환경(녹색)에서 실행합니다. 이 전략을 사용하면 영향을 최소화하면서 신속하게 롤백할 수 있습니다.

bot

인터넷을 통해 자동화된 태스크를 실행하고 인적 활동이나 상호 작용을 시뮬레이션하는 소프트웨어 애플리케이션입니다. 인터넷에서 정보를 인덱싱하는 웹 크롤러와 같이 유용하거나 이로운 봇도 있습니다. 악성 봇이라고 하는 다른 일부 봇은 개인 또는 조직을 방해하거나 해를 입히기 위한 봇입니다.

봇넷

맬웨어에 감염되고 봇 허더 또는 봇 운영자와 같은 단일 당사자가 제어하는 봇 네트워크입니다. 봇넷은 봇의 규모와 봇의 영향 범위를 확대하는 가장 잘 알려진 메커니즘입니다.

브랜치

코드 리포지토리의 포함된 영역입니다. 리포지토리에 생성되는 첫 번째 브랜치가 기본 브랜치입니다. 기존 브랜치에서 새 브랜치를 생성한 다음 새 브랜치에서 기능을 개발하거나 버그를 수정할 수 있습니다. 기능을 구축하기 위해 생성하는 브랜치를 일반적으로 기능 브랜치라고 합니다. 기능을 출시할 준비가 되면 기능 브랜치를 기본 브랜치에 다시 병합합니다. 자세한 내용은 About branches(GitHub 설명서)를 참조하십시오.

긴급 액세스 권한

예외적인 상황에서 승인된 프로세스를 통해 사용자가 일반적으로 액세스할 권한이 없는에 액세스할 수 AWS 계정 있는 빠른 방법입니다. 자세한 내용은 AWS Well-Architected 지침의 Implement break-glass procedures 지표를 참조하세요.

브라운필드 전략

사용자 환경의 기존 인프라 시스템 아키텍처에 브라운필드 전략을 채택할 때는 현재 시스템 및 인프라의 제약 조건을 중심으로 아키텍처를 설계합니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

버퍼 캐시

가장 자주 액세스하는 데이터가 저장되는 메모리 영역입니다.

사업 역량

기업이 가치를 창출하기 위해 하는 일(예: 영업, 고객 서비스 또는 마케팅)입니다. 마이크로서비스 아키텍처 및 개발 결정은 비즈니스 역량에 따라 이루어질 수 있습니다. 자세한 내용은 백서의 AWS에서 컨테이너화된 마이크로서비스 실행의 비즈니스 역량 중심의 구성화 섹션을 참조하십시오.

비즈니스 연속성 계획(BCP)

대규모 마이그레이션과 같은 중단 이벤트가 운영에 미치는 잠재적 영향을 해결하고 비즈니스가 신속하게 운영을 재개할 수 있도록 지원하는 계획입니다.

C

CAF

AWS Cloud Adoption Framework를 참조하세요.

카나리 배포

최종 사용자에게 제공하는 느린 증분 릴리스 버전입니다. 확신이 들면 새 버전을 배포하고 현재 버전을 완전히 교체합니다.

CCoE

클라우드 혁신 센터를 참조하세요.

CDC

데이터 캡처 변경을 참조하세요.

변경 데이터 캡처(CDC)

데이터베이스 테이블과 같은 데이터 소스의 변경 내용을 추적하고 변경 사항에 대한 메타데이터를 기록하는 프로세스입니다. 대상 시스템의 변경 내용을 감사하거나 복제하여 동기화를 유지하는 등의 다양한 용도로 CDC를 사용할 수 있습니다.

카오스 엔지니어링

시스템의 복원력을 테스트하기 위해 의도적으로 장애나 중단 이벤트를 도입합니다. AWS Fault Injection Service (AWS FIS)를 사용하여 AWS 워크로드에 스트레스를 주고 응답을 평가하는 실험을 수행할 수 있습니다.

CI/CD

지속적 통합 및 지속적 전송을 참조하세요.

분류

예측을 생성하는 데 도움이 되는 분류 프로세스입니다. 분류 문제에 대한 ML 모델은 이산 값을 예측합니다. 이산 값은 항상 서로 다릅니다. 예를 들어, 모델이 이미지에 자동차가 있는지 여부를 평가해야 할 수 있습니다.

클라이언트측 암호화

대상이 데이터를 AWS 서비스 수신하기 전에 로컬에서 데이터를 암호화합니다.

클라우드 혁신 센터(CCoE)

클라우드 모범 사례 개발, 리소스 동원, 마이그레이션 타임라인 설정, 대규모 혁신을 통한 조직 선도 등 조직 전체에서 클라우드 채택 노력을 추진하는 다분야 팀입니다. 자세한 내용은 AWS 클라우드 엔터프라이즈 전략 블로그의 CCoE 게시물을 참조하세요.

클라우드 컴퓨팅

원격 데이터 스토리지와 IoT 디바이스 관리에 일반적으로 사용되는 클라우드 기술 클라우드 컴퓨팅은 일반적으로 엣지 컴퓨팅 기술에 연결되어 있습니다.

클라우드 운영 모델

IT 조직에서 하나 이상의 클라우드 환경을 구축, 성숙화 및 최적화하는 데 사용되는 운영 모델입니다. 자세한 내용은 클라우드 운영 모델 구축을 참조하십시오.

클라우드 채택 단계

조직이 AWS 클라우드로 마이그레이션할 때 일반적으로 거치는 4단계는 다음과 같습니다.

- 프로젝트 - 개념 증명 및 학습 목적으로 몇 가지 클라우드 관련 프로젝트 실행
- 기반 – 클라우드 채택 확장을 위한 기초 투자(예: 랜딩 존 생성, CCoE 정의, 운영 모델 구축)
- 마이그레이션 - 개별 애플리케이션 마이그레이션
- Re-invention - 제품 및 서비스 최적화와 클라우드 혁신

이러한 단계는 Stephen Orban이 블로그 게시물 The Journey Toward Cloud-First and the Stages of Adoption on the AWS 클라우드 Enterprise Strategy 블로그에서 정의했습니다. AWS 마이그레이션 전략과 어떤 관련이 있는지에 대한 자세한 내용은 마이그레이션 준비 가이드를 참조하세요.

CMDB

구성 관리 데이터베이스를 참조하세요.

코드 리포지토리

소스 코드와 설명서, 샘플, 스크립트 등의 기타 자산이 버전 관리 프로세스를 통해 저장되고 업데이트되는 위치입니다. 일반적인 클라우드 리포지토리로 GitHub 또는 Bitbucket Cloud가 포함됩니다. 코드의 각 버전을 브랜치라고 합니다. 마이크로서비스 구조에서 각 리포지토리는 단일 기능 전용입니다. 단일 CI/CD 파이프라인은 여러 리포지토리를 사용할 수 있습니다.

콜드 캐시

비어 있거나, 제대로 채워지지 않았거나, 오래되었거나 관련 없는 데이터를 포함하는 버퍼 캐시입니다. 주 메모리나 디스크에서 데이터베이스 인스턴스를 읽어야 하기 때문에 성능에 영향을 미치며, 이는 버퍼 캐시에서 읽는 것보다 느립니다.

콜드 데이터

거의 액세스되지 않고 일반적으로 과거 데이터인 데이터. 이런 종류의 데이터를 쿼리할 때는 일반적으로 느린 쿼리가 허용됩니다. 이 데이터를 성능이 낮고 비용이 저렴한 스토리지 계층 또는 클래스로 옮기면 비용을 절감할 수 있습니다.

컴퓨터 비전(CV)

기계 학습을 사용하여 디지털 이미지 및 비디오와 같은 시각적 형식에서 정보를 분석하고 추출하는 AI 필드입니다. 예를 들어 Amazon SageMaker AI는 CV에 대한 이미지 처리 알고리즘을 제공합니다.

구성 드리프트

워크로드의 경우 구성이 예상되는 상태에서 변경됩니다. 이로 인해 워크로드가 규정을 준수하지 않을 수 있으며, 이는 일반적으로 점진적이고 의도되지 않은 작업입니다.

구성 관리 데이터베이스(CMDB)

하드웨어 및 소프트웨어 구성 요소와 해당 구성을 포함하여 데이터베이스와 해당 IT 환경에 대한 정보를 저장하고 관리하는 리포지토리입니다. 일반적으로 마이그레이션의 포트폴리오 탐색 및 분석 단계에서 CMDB의 데이터를 사용합니다.

규정 준수 팩

규정 준수 및 보안 검사를 사용자 지정하기 위해 조합할 수 있는 AWS Config 규칙 및 문제 해결 작업의 모음입니다. YAML 템플릿을 사용하여 적합성 팩을 AWS 계정 및 리전 또는 조직 전체에 단일 엔터티로 배포할 수 있습니다. 자세한 내용은 AWS Config 설명서의 적합성 팩을 참조하세요.

지속적 통합 및 지속적 전달(CI/CD)

소프트웨어 릴리스 프로세스의 소스, 빌드, 테스트, 스테이징 및 프로덕션 단계를 자동화하는 프로세스입니다. CI/CD는 일반적으로 파이프라인으로 설명됩니다. CI/CD를 통해 프로세스를 자동화하고, 생산성을 높이고, 코드 품질을 개선하고, 더 빠르게 제공할 수 있습니다. 자세한 내용은 지속적 전달의 이점을 참조하십시오. CD는 지속적 배포를 의미하기도 합니다. 자세한 내용은 지속적 전달(Continuous Delivery)과 지속적인 개발을 참조하십시오.

CV

컴퓨터 비전을 참조하세요.

D

저장 데이터

스토리지에 있는 데이터와 같이 네트워크에 고정되어 있는 데이터입니다.

데이터 분류

중요도와 민감도를 기준으로 네트워크의 데이터를 식별하고 분류하는 프로세스입니다. 이 프로세스는 데이터에 대한 적절한 보호 및 보존 제어를 결정하는 데 도움이 되므로 사이버 보안 위험 관리 전략의 중요한 구성 요소입니다. 데이터 분류는 AWS Well-Architected Framework에서 보안 원칙의 구성 요소입니다. 자세한 내용은 데이터 분류를 참조하십시오.

데이터 드리프트

프로덕션 데이터와 ML 모델 학습에 사용된 데이터 간의 상당한 차이 또는 시간 경과에 따른 입력 데이터의 의미 있는 변화. 데이터 드리프트는 ML 모델 예측의 전반적인 품질, 정확성 및 공정성을 저하시킬 수 있습니다.

전송 중 데이터

네트워크를 통과하고 있는 데이터입니다. 네트워크 리소스 사이를 이동 중인 데이터를 예로 들 수 있습니다.

데이터 메시

중앙 집중식 관리 및 거버넌스를 통해 분산되고 탈중앙화된 데이터 소유권을 제공하는 아키텍처 프레임워크입니다.

데이터 최소화

꼭 필요한 데이터만 수집하고 처리하는 원칙입니다. 에서 데이터를 최소화하면 개인 정보 보호 위험, 비용 및 분석 탄소 발자국을 줄일 AWS 클라우드 수 있습니다.

데이터 경계

신뢰할 수 있는 자격 증명만 예상 네트워크에서 신뢰할 수 있는 리소스에 액세스하도록 하는 데 도움이 되는 AWS 환경의 예방 가드레일 세트입니다. 자세한 내용은 데이터 경계 구축을 참조하세요 AWS.

데이터 사전 처리

원시 데이터를 ML 모델이 쉽게 구문 분석할 수 있는 형식으로 변환하는 것입니다. 데이터를 사전 처리한다는 것은 특정 열이나 행을 제거하고 누락된 값, 일관성이 없는 값 또는 중복 값을 처리함을 의미할 수 있습니다.

데이터 출처

라이프사이클 전반에 걸쳐 데이터의 출처와 기록을 추적하는 프로세스(예: 데이터 생성, 전송, 저장 방법).

데이터 주체

데이터를 수집 및 처리하는 개인입니다.

데이터 웨어하우스

분석과 같은 비즈니스 인텔리전스를 지원하는 데이터 관리 시스템입니다. 데이터 웨어하우스에는 보통 많은 양의 기록 데이터가 포함되며 일반적으로 쿼리 및 분석에 사용됩니다.

데이터 정의 언어(DDL)

데이터베이스에서 테이블 및 객체의 구조를 만들거나 수정하기 위한 명령문 또는 명령입니다.

데이터베이스 조작 언어(DML)

데이터베이스에서 정보를 수정(삽입, 업데이트 및 삭제)하기 위한 명령문 또는 명령입니다.

DDL

데이터 정의 언어를 참조하세요.

딥 앙상블

예측을 위해 여러 딥 러닝 모델을 결합하는 것입니다. 딥 앙상블을 사용하여 더 정확한 예측을 얻거나 예측의 불확실성을 추정할 수 있습니다.

딥 러닝

여러 계층의 인공 신경망을 사용하여 입력 데이터와 관심 대상 변수 간의 매핑을 식별하는 ML 하위 분야입니다.

심층 방어

네트워크와 그 안의 데이터 기밀성, 무결성 및 가용성을 보호하기 위해 컴퓨터 네트워크 전체에 일련의 보안 메커니즘과 제어를 신중하게 계층화하는 정보 보안 접근 방식입니다. 이 전략을 채택하면 AWS Organizations 구조의 여러 계층에 여러 제어를 AWS추가하여 리소스를 보호할 수 있습니다. 예를 들어, 심층 방어 접근 방식은 다단계 인증, 네트워크 세분화 및 암호화를 결합할 수 있습니다.

위임된 관리자

에서 AWS Organizations호환되는 서비스는 AWS 멤버 계정을 등록하여 조직의 계정을 관리하고 해당 서비스에 대한 권한을 관리할 수 있습니다. 이러한 계정을 해당 서비스의 위임된 관리자라고 합니다. 자세한 내용과 호환되는 서비스 목록은 AWS Organizations 설명서의 AWS Organizations와 함께 사용할 수 있는 AWS 서비스를 참조하십시오.

배포

대상 환경에서 애플리케이션, 새 기능 또는 코드 수정 사항을 사용할 수 있도록 하는 프로세스입니다. 배포에는 코드 베이스의 변경 사항을 구현한 다음 애플리케이션 환경에서 해당 코드베이스를 구축하고 실행하는 작업이 포함됩니다.

개발 환경

환경을 참조하세요.

탐지 제어

이벤트 발생 후 탐지, 기록 및 알림을 수행하도록 설계된 보안 제어입니다. 이러한 제어는 기존의 예방적 제어를 우회한 보안 이벤트를 알리는 2차 방어선입니다. 자세한 내용은 AWS에서 보안 제어 구현의 탐지 제어를 참조하세요.

개발 가치 흐름 매핑 (DVSM)

소프트웨어 개발 라이프사이클에서 속도와 품질에 부정적인 영향을 미치는 제약 조건을 식별하고 우선 순위를 지정하는 데 사용되는 프로세스입니다. DVSM은 원래 린 제조 방식을 위해 설계된 가치 흐름 매핑 프로세스를 확장합니다. 소프트웨어 개발 프로세스를 통해 가치를 창출하고 이동하는 데 필요한 단계와 팀에 중점을 둡니다.

디지털 트윈

건물, 공장, 산업 장비 또는 생산 라인과 같은 실제 시스템을 가상으로 표현한 것입니다. 디지털 트윈은 예측 유지 보수, 원격 모니터링, 생산 최적화를 지원합니다.

차원 테이블

스타 스키마에서 팩트 테이블의 정량적 데이터에 대한 데이터 속성을 포함하는 더 작은 테이블을 말합니다. 차원 테이블 속성은 일반적으로 텍스트 필드나 텍스트처럼 동작하는 개별 숫자입니다. 이러한 속성은 보통 쿼리 제약, 필터링 및 결과 세트 레이블 지정에 사용됩니다.

재해

워크로드 또는 시스템이 기본 배포 위치에서 비즈니스 목표를 달성하지 못하게 방해하는 이벤트입니다. 이러한 이벤트는 자연재해, 기술적 오류, 의도하지 않은 구성 오류 또는 멀웨어 공격과 같은 사람의 행동으로 인한 결과일 수 있습니다.

재해 복구(DR)

재해로 인한 가동 중지 시간 및 데이터 손실을 최소화하기 위해 사용하는 전략 및 프로세스입니다. 자세한 내용은 AWS Well-Architected Framework의 Disaster Recovery of Workloads on AWS: Recovery in the Cloud를 참조하세요.

DML

데이터베이스 조작 언어를 참조하세요.

도메인 기반 설계

구성 요소를 각 구성 요소가 제공하는 진화하는 도메인 또는 핵심 비즈니스 목표에 연결하여 복잡한 소프트웨어 시스템을 개발하는 접근 방식입니다. 이 개념은 에릭 에반스에 의해 그의 저서인 도메인 기반 디자인: 소프트웨어 중심의 복잡성 해결(Boston: Addison-Wesley Professional, 2003)에서 소개되었습니다. Strangler Fig 패턴과 함께 도메인 기반 설계를 사용하는 방법에 대한 자세한 내용은 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

DR

재해 복구를 참조하세요.

드리프트 감지

기준이 되는 구성과의 편차 추적을 말합니다. 예를 들어 AWS CloudFormation 를 사용하여 시스템 리소스의 드리프트를 감지하거나를 사용하여 AWS Control Tower 거버넌스 요구 사항 준수에 영향을 미칠 수 있는 랜딩 존의 변경 사항을 감지할 수 있습니다.

DVSM

개발 가치 흐름 매핑을 참조하세요.

E

EDA

탐색 데이터 분석을 참조하세요.

EDI

전자 데이터 교환을 참조하세요.

엣지 컴퓨팅

IoT 네트워크의 엣지에서 스마트 디바이스의 컴퓨팅 성능을 개선하는 기술 엣지 컴퓨팅은 클라우드 컴퓨팅에 비해 보다 통신 지연 시간을 줄이고 응답 시간을 개선할 수 있습니다.

전자 데이터 교환(EDI)

조직 간 비즈니스 문서의 자동화된 교환을 나타냅니다. 자세한 내용은 전자 데이터 교환(EDI)이란 무엇인가요?를 참조하세요.

암호화

사람이 읽을 수 있는 일반 텍스트 데이터를 사이퍼텍스트로 변환하는 컴퓨팅 프로세스입니다.

암호화 키

암호화 알고리즘에 의해 생성되는 무작위 비트의 암호화 문자열입니다. 키의 길이는 다양할 수 있으며 각 키는 예측할 수 없고 고유하게 설계되었습니다.

엔디안

컴퓨터 메모리에 바이트가 저장되는 순서입니다. 빅 엔디안 시스템은 가장 중요한 바이트를 먼저 저장합니다. 리틀 엔디안 시스템은 가장 덜 중요한 바이트를 먼저 저장합니다.

엔드포인트

서비스 엔드포인트를 참조하세요.

엔드포인트 서비스

Virtual Private Cloud(VPC)에서 호스팅하여 다른 사용자와 공유할 수 있는 서비스입니다. 를 사용하여 엔드포인트 서비스를 생성하고 다른 AWS 계정 또는 AWS Identity and Access Management (IAM) 보안 주체에 권한을 AWS PrivateLink 부여할 수 있습니다. 이러한 계정 또는 보안 주체는 인터페이스 VPC 엔드포인트를 생성하여 엔드포인트 서비스에 비공개로 연결할 수 있습니다. 자세한 내용은 Amazon Virtual Private Cloud(VPC) 설명서의 엔드포인트 서비스 생성을 참조하십시오.

엔터프라이즈 리소스 계획(ERP)

엔터프라이즈의 주요 비즈니스 프로세스(예: 회계, MES, 프로젝트 관리)를 자동화하고 관리하는 시스템입니다.

봉투 암호화

암호화 키를 다른 암호화 키로 암호화하는 프로세스입니다. 자세한 내용은 AWS Key Management Service (AWS KMS) 설명서의 봉투 암호화를 참조하세요.

환경

실행 중인 애플리케이션의 인스턴스입니다. 다음은 클라우드 컴퓨팅의 일반적인 환경 유형입니다.

- 개발 환경 - 애플리케이션 유지 관리를 담당하는 핵심 팀만 사용할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. 개발 환경은 변경 사항을 상위 환경으로 승격하기 전에 테스트하는 데 사용됩니다. 이러한 유형의 환경을 테스트 환경이라고도 합니다.
- 하위 환경 - 초기 빌드 및 테스트에 사용되는 환경을 비롯한 애플리케이션의 모든 개발 환경입니다.
- 프로덕션 환경 - 최종 사용자가 액세스할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. CI/CD 파이프라인에서 프로덕션 환경이 마지막 배포 환경입니다.
- 상위 환경 - 핵심 개발 팀 이외의 사용자가 액세스할 수 있는 모든 환경입니다. 프로덕션 환경, 프로덕션 이전 환경 및 사용자 수용 테스트를 위한 환경이 여기에 포함될 수 있습니다.

에픽

애자일 방법론에서 작업을 구성하고 우선순위를 정하는 데 도움이 되는 기능적 범주입니다. 에픽은 요구 사항 및 구현 작업에 대한 개괄적인 설명을 제공합니다. 예를 들어, AWS CAF 보안 에픽에는 ID 및 액세스 관리, 탐지 제어, 인프라 보안, 데이터 보호 및 인시던트 대응이 포함됩니다. AWS 마이그레이션 전략의 에픽에 대한 자세한 내용은 프로그램 구현 가이드를 참조하십시오.

ERP

엔터프라이즈 리소스 계획을 참조하세요.

탐색 데이터 분석(EDA)

데이터 세트를 분석하여 주요 특성을 파악하는 프로세스입니다. 데이터를 수집 또는 집계한 다음 초기 조사를 수행하여 패턴을 찾고, 이상을 탐지하고, 가정을 확인합니다. EDA는 요약 통계를 계산하고 데이터 시각화를 생성하여 수행됩니다.

F

팩트 테이블

스타 스키마의 중앙 테이블입니다. 비즈니스 운영에 대한 정량적 데이터를 저장합니다. 일반적으로 팩트 테이블은 측정값이 있는 열 및 차원 테이블에 대한 외래 키가 있는 열과 같이 두 가지 열 유형을 포함합니다.

빠른 실패

개발 수명 주기를 줄이기 위해 빈번한 증분 테스트를 사용하는 철학입니다. 애자일 접근 방식의 핵심입니다.

장애 격리 경계

에서 장애의 영향을 제한하고 워크로드의 복원력을 개선하는 데 도움이 되는 가용 영역, AWS 리전 컨트롤 플레인 또는 데이터 플레인과 같은 AWS 클라우드경계입니다. 자세한 내용은 AWS 장애 격리 경계를 참조하세요.

기능 브랜치

브랜치를 참조하세요.

기능

예측에 사용하는 입력 데이터입니다. 예를 들어, 제조 환경에서 기능은 제조 라인에서 주기적으로 캡처되는 이미지일 수 있습니다.

기능 중요도

모델의 예측에 특성이 얼마나 중요한지를 나타냅니다. 이는 일반적으로 SHAP(Shapley Additive Descriptions) 및 통합 그레디언트와 같은 다양한 기법을 통해 계산할 수 있는 수치 점수로 표현됩니다. 자세한 내용은 기계 학습 모델 해석 가능성을 참조하세요 AWS.

기능 변환

추가 소스로 데이터를 보강하거나, 값을 조정하거나, 단일 데이터 필드에서 여러 정보 세트를 추출하는 등 ML 프로세스를 위해 데이터를 최적화하는 것입니다. 이를 통해 ML 모델이 데이터를 활용

할 수 있습니다. 예를 들어, 날짜 '2021-05-27 00:15:37'을 '2021년', '5월', '목', '15일'로 분류하면 학습 알고리즘이 다양한 데이터 구성 요소와 관련된 미묘한 패턴을 학습하는 데 도움이 됩니다.

퓨샷 프롬프팅

유사한 태스크를 수행하도록 요청하기 전에 LLM에 태스크와 원하는 출력을 보여주는 몇 가지 예제를 제공합니다. 이 기법은 모델이 프롬프트에 포함된 예제(샷)에서 학습하는 컨텍스트 내 학습을 적용합니다. 퓨샷 프롬프팅은 특정 형식 지정, 추론 또는 분야별 지식이 필요한 태스크에 효과적일 수 있습니다. 제로샷 프롬프팅도 참조하세요.

FGAC

세분화된 액세스 제어를 참조하세요.

세분화된 액세스 제어(FGAC)

여러 조건을 사용하여 액세스 요청을 허용하거나 거부합니다.

플래시컷 마이그레이션

단계적 접근 방식을 사용하는 대신 변경 데이터 캡처를 통해 지속적 데이터 복제를 사용하여 최단 시간에 데이터를 마이그레이션하는 데이터베이스 마이그레이션 방법입니다. 목표는 가동 중지 시간을 최소화하는 것입니다.

FM

파운데이션 모델을 참조하세요.

파운데이션 모델(FM)

일반화되고 레이블이 지정되지 않은 데이터의 대규모 데이터세트에서 훈련된 대규모 딥 러닝 신경망입니다. FM은 언어 이해, 텍스트 및 이미지 생성, 자연어 대화와 같은 다양한 일반 태스크를 수행할 수 있습니다. 자세한 내용은 파운데이션 모델이란?을 참조하세요.

G

생성형 AI

대량의 데이터에서 훈련되었으며 간단한 텍스트 프롬프트를 사용하여 이미지, 비디오, 텍스트, 오디오와 같은 새 콘텐츠와 아티팩트를 생성할 수 있는 AI 모델의 하위 세트입니다. 자세한 내용은 생성형 AI란 무엇인가요?를 참조하세요.

지리적 차단

지리적 제한을 참조하세요.

지리적 제한(지리적 차단)

Amazon CloudFront에서 특정 국가의 사용자가 콘텐츠 배포에 액세스하지 못하도록 하는 옵션입니다. 허용 목록 또는 차단 목록을 사용하여 승인된 국가와 차단된 국가를 지정할 수 있습니다. 자세한 내용은 CloudFront 설명서의 콘텐츠의 지리적 배포 제한을 참조하십시오.

Gitflow 워크플로

하위 환경과 상위 환경이 소스 코드 리포지토리의 서로 다른 브랜치를 사용하는 방식입니다. Gitflow 워크플로는 레거시로 간주되며 트렁크 기반 워크플로는 선호되는 현대적 접근 방식입니다.

골든 이미지

시스템 또는 소프트웨어의 새 인스턴스를 배포하기 위한 템플릿으로 사용되는 해당 시스템 또는 소프트웨어의 스냅샷입니다. 예를 들어 제조 분야에서는 골든 이미지를 사용하여 여러 디바이스에서 소프트웨어를 프로비저닝할 수 있으며 이를 통해 딥이스 제조 작업의 속도, 확장성 및 생산성을 개선할 수 있습니다.

브라운필드 전략

새로운 환경에서 기존 인프라의 부재 시스템 아키텍처에 대한 그린필드 전략을 채택할 때 브라운필드라고도 하는 기존 인프라와의 호환성 제한 없이 모든 새로운 기술을 선택할 수 있습니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

가드레일

조직 단위(OU) 전체에서 리소스, 정책 및 규정 준수를 관리하는 데 도움이 되는 중요 규칙입니다. 예방 가드레일은 규정 준수 표준에 부합하도록 정책을 시행하며, 서비스 제어 정책과 IAM 권한 경계를 사용하여 구현됩니다. 탐지 가드레일은 정책 위반 및 규정 준수 문제를 감지하고 해결을 위한 알림을 생성하며, 이는 AWS Config Amazon GuardDuty AWS Security Hub CSPM, , AWS Trusted Advisor Amazon Inspector 및 사용자 지정 AWS Lambda 검사를 사용하여 구현됩니다.

H

HA

고가용성을 참조하세요.

이기종 데이터베이스 마이그레이션

다른 데이터베이스 엔진을 사용하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Oracle에서 Amazon Aurora로) 이기종 마이그레이션은 일반적으로 리아키텍트 작업의 일부이며 스

키마를 변환하는 것은 복잡한 작업일 수 있습니다.AWS 는 스키마 변환에 도움이 되는 AWS SCT를 제공합니다.

높은 가용성(HA)

문제나 재해 발생 시 개입 없이 지속적으로 운영할 수 있는 워크로드의 능력. HA 시스템은 자동으로 장애 조치되고, 지속적으로 고품질 성능을 제공하고, 성능에 미치는 영향을 최소화하면서 다양한 부하와 장애를 처리하도록 설계되었습니다.

히스토리언 현대화

제조 산업의 요구 사항을 더 잘 충족하도록 운영 기술(OT) 시스템을 현대화하고 업그레이드하는 데 사용되는 접근 방식입니다. 히스토리언은 공장의 다양한 출처에서 데이터를 수집하고 저장하는 데 사용되는 일종의 데이터베이스입니다.

홀드아웃 데이터

기계 학습 모델을 훈련하는 데 사용되는 데이터세트에서 보류되는 레이블이 지정된 기록 데이터의 일부입니다. 홀드아웃 데이터를 사용하여 모델 예측을 홀드아웃 데이터와 비교해 모델 성능을 평가할 수 있습니다.

동종 데이터베이스 마이그레이션

동일한 데이터베이스 엔진을 공유하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Microsoft SQL Server에서 Amazon RDS for SQL Server로) 동종 마이그레이션은 일반적으로 리호스팅 또는 리플랫포밍 작업의 일부입니다. 네이티브 데이터베이스 유틸리티를 사용하여 스키마를 마이그레이션할 수 있습니다.

핫 데이터

자주 액세스하는 데이터(예: 실시간 데이터 또는 최근 번역 데이터). 일반적으로 이 데이터에는 빠른 쿼리 응답을 제공하기 위한 고성능 스토리지 계층 또는 클래스가 필요합니다.

핫픽스

프로덕션 환경의 중요한 문제를 해결하기 위한 긴급 수정입니다. 핫픽스는 긴급하기 때문에 일반적인 DevOps 릴리스 워크플로 외부에서 실행됩니다.

하이퍼케어 기간

전환 직후 마이그레이션 팀이 문제를 해결하기 위해 클라우드에서 마이그레이션된 애플리케이션을 관리하고 모니터링하는 기간입니다. 일반적으로 이 기간은 1~4일입니다. 하이퍼케어 기간이 끝나면 마이그레이션 팀은 일반적으로 애플리케이션에 대한 책임을 클라우드 운영 팀에 넘깁니다.

I

IaC

코드형 인프라를 참조하세요.

자격 증명 기반 정책

AWS 클라우드 환경 내에서 권한을 정의하는 하나 이상의 IAM 보안 주체에 연결된 정책입니다.

유휴 애플리케이션

90일 동안 평균 CPU 및 메모리 사용량이 5~20%인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하거나 온프레미스에 유지하는 것이 일반적입니다.

IIoT

산업용 사물 인터넷을 참조하세요.

변경 불가능한 인프라

기존 인프라를 업데이트, 패치 또는 수정하는 대신 프로덕션 워크로드에 대한 새 인프라를 배포하는 모델입니다. 변경 불가능한 인프라는 변경 가능한 인프라보다 본질적으로 더 일관되고 안정적이며 예측 가능합니다. 자세한 내용은 AWS Well-Architected Framework의 변경 불가능한 인프라를 사용하여 배포 모범 사례를 참조하세요.

인바운드(수신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 외부에서 네트워크 연결을 수락, 검사 및 라우팅하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

증분 마이그레이션

한 번에 전체 전환을 수행하는 대신 애플리케이션을 조금씩 마이그레이션하는 전환 전략입니다. 예를 들어, 처음에는 소수의 마이크로서비스나 사용자만 새 시스템으로 이동할 수 있습니다. 모든 것이 제대로 작동하는지 확인한 후에는 레거시 시스템을 폐기할 수 있을 때까지 추가 마이크로서비스 또는 사용자를 점진적으로 이동할 수 있습니다. 이 전략을 사용하면 대규모 마이그레이션과 관련된 위험을 줄일 수 있습니다.

Industry 4.0

연결성, 실시간 데이터, 자동화, 분석 및 AI/ML의 발전을 통해 제조 프로세스의 현대화를 나타내기 위해 2016년에 Klaus Schwab에서 도입한 용어입니다.

인프라

애플리케이션의 환경 내에 포함된 모든 리소스와 자산입니다.

코드형 인프라(IaC)

구성 파일 세트를 통해 애플리케이션의 인프라를 프로비저닝하고 관리하는 프로세스입니다. IaC는 새로운 환경의 반복 가능성, 신뢰성 및 일관성을 위해 인프라 관리를 중앙 집중화하고, 리소스를 표준화하고, 빠르게 확장할 수 있도록 설계되었습니다.

산업용 사물 인터넷(IIoT)

제조, 에너지, 자동차, 의료, 생명과학, 농업 등의 산업 부문에서 인터넷에 연결된 센서 및 디바이스의 사용 자세한 내용은 산업용 사물 인터넷(IoT) 디지털 트랜스포메이션 전략 구축을 참조하십시오.

검사 VPC

AWS 다중 계정 아키텍처에서는 VPC(동일하거나 다른 AWS 리전), 인터넷 및 온프레미스 네트워크 간의 네트워크 트래픽 검사를 관리하는 중앙 집중식 VPCs입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

사물 인터넷(IoT)

인터넷이나 로컬 통신 네트워크를 통해 다른 디바이스 및 시스템과 통신하는 센서 또는 프로세서가 내장된 연결된 물리적 객체의 네트워크 자세한 내용은 IoT란?을 참조하십시오.

해석력

모델의 예측이 입력에 따라 어떻게 달라지는지를 사람이 이해할 수 있는 정도를 설명하는 기계 학습 모델의 특성입니다. 자세한 내용은 기계 학습 모델 해석 가능성을 참조하세요 AWS.

IoT

사물 인터넷을 참조하세요.

IT 정보 라이브러리(ITIL)

IT 서비스를 제공하고 이러한 서비스를 비즈니스 요구 사항에 맞게 조정하기 위한 일련의 모범 사례 ITIL은 ITSM의 기반을 제공합니다.

IT 서비스 관리(ITSM)

조직의 IT 서비스 설계, 구현, 관리 및 지원과 관련된 활동 클라우드 운영을 ITSM 도구와 통합하는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

ITIL

IT 정보 라이브러리를 참조하세요.

ITSM

IT 서비스 관리를 참조하세요.

L

레이블 기반 액세스 제어(LBAC)

사용자 및 데이터 자체에 각각 보안 레이블 값을 명시적으로 할당하는 필수 액세스 제어(MAC)를 구현한 것입니다. 사용자 보안 레이블과 데이터 보안 레이블 간의 교차 부분에 따라 사용자가 볼 수 있는 행과 열이 결정됩니다.

랜딩 존

랜딩 존은 확장 가능하고 안전한 잘 설계된 다중 계정 AWS 환경입니다. 조직은 여기에서부터 보안 및 인프라 환경에 대한 확신을 가지고 워크로드와 애플리케이션을 신속하게 시작하고 배포할 수 있습니다. 랜딩 존에 대한 자세한 내용은 안전하고 확장 가능한 다중 계정 AWS 환경 설정을 참조하십시오.

대규모 언어 모델(LLM)

방대한 양의 데이터에서 사전 훈련된 딥 러닝 AI 모델입니다. LLM은 질문에 대한 답변, 문서 요약, 텍스트를 다른 언어로 번역, 문장 완성과 같은 여러 태스크를 수행할 수 있습니다. 자세한 내용은 대규모 언어 모델(LLM)이란 무엇인가요?를 참조하세요.

대규모 마이그레이션

300대 이상의 서버 마이그레이션입니다.

LBAC

레이블 기반 액세스 제어를 참조하세요.

최소 권한

작업을 수행하는 데 필요한 최소 권한을 부여하는 보안 모범 사례입니다. 자세한 내용은 IAM 설명서의 최소 권한 적용을 참조하십시오.

리프트 앤드 시프트

7R을 참조하세요.

리틀 엔디안 시스템

가장 덜 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안도 참조하세요.

LLM

대규모 언어 모델을 참조하세요.

하위 환경

환경을 참조하세요.

M

기계 학습(ML)

패턴 인식 및 학습에 알고리즘과 기법을 사용하는 인공 지능의 한 유형입니다. ML은 사물 인터넷(IoT) 데이터와 같은 기록된 데이터를 분석하고 학습하여 패턴을 기반으로 통계 모델을 생성합니다. 자세한 내용은 기계 학습을 참조하십시오.

기본 브랜치

브랜치를 참조하세요.

맬웨어

컴퓨터 보안 또는 프라이버시를 위협하도록 설계된 소프트웨어입니다. 맬웨어는 컴퓨터 시스템을 방해하거나 민감한 정보를 유출하거나 무단 액세스 권한을 확보할 수 있습니다. 맬웨어의 예로 바이러스, 웜, 랜섬웨어, 트로이 목마, 스파이웨어, 키로거 등이 있습니다.

관리형 서비스

AWS 서비스 는 인프라 계층, 운영 체제 및 플랫폼을 AWS 운영하고, 사용자는 엔드포인트에 액세스하여 데이터를 저장하고 검색합니다. 관리형 서비스의 예로 Amazon Simple Storage Service(Amazon S3) 및 Amazon DynamoDB가 있습니다. 이를 추상화된 서비스라고도 합니다.

제조 실행 시스템(MES)

원자재를 생산 현장에서 완제품으로 변환하는 생산 프로세스를 추적, 모니터링, 문서화 및 제어하기 위한 소프트웨어 시스템입니다.

MAP

Migration Acceleration Program을 참조하세요.

메커니즘

도구를 생성하고 도구 채택을 유도한 다음 조정을 위해 결과를 검사하는 전체 프로세스입니다. 메커니즘은 작동 시 자체적으로 강화하고 개선하는 주기입니다. 자세한 내용은 AWS Well-Architected Framework의 메커니즘 구축을 참조하세요.

멤버 계정

조직의 일부인 관리 계정을 AWS 계정 제외한 모든 계정. AWS Organizations하나의 계정은 한 번에 하나의 조직 멤버만 될 수 있습니다.

MES

제조 실행 시스템을 참조하세요.

메시지 큐 원격 분석 전송(MQTT)

리소스 제약이 있는 IoT 디바이스에 대한 게시 및 구독 패턴을 기반으로 하는 경량 Machine-to-Machine(M2M) 통신 프로토콜입니다.

마이크로서비스

잘 정의된 API를 통해 통신하고 일반적으로 소규모 자체 팀이 소유하는 소규모 독립 서비스입니다. 예를 들어, 보험 시스템에는 영업, 마케팅 등의 비즈니스 역량이나 구매, 청구, 분석 등의 하위 영역에 매핑되는 마이크로 서비스가 포함될 수 있습니다. 마이크로서비스의 이점으로 민첩성, 유연한 확장, 손쉬운 배포, 재사용 가능한 코드, 복원력 등이 있습니다. 자세한 내용은 AWS 서버리스 서비스를 사용하여 마이크로서비스 통합을 참조하세요.

마이크로서비스 아키텍처

각 애플리케이션 프로세스를 마이크로서비스로 실행하는 독립 구성 요소를 사용하여 애플리케이션을 구축하는 접근 방식입니다. 이러한 마이크로서비스는 경량 API를 사용하여 잘 정의된 인터페이스를 통해 통신합니다. 애플리케이션의 특정 기능에 대한 수요에 맞게 이 아키텍처의 각 마이크로서비스를 업데이트, 배포 및 조정할 수 있습니다. 자세한 내용은 에서 마이크로서비스 구현을 참조하세요 AWS.

Migration Acceleration Program(MAP)

조직이 클라우드로 전환하기 위한 강력한 운영 기반을 구축하고 초기 마이그레이션 비용을 상쇄하는 데 도움이 되는 컨설팅 지원, 교육 및 서비스를 제공하는 AWS 프로그램입니다. MAP에는 레거시 마이그레이션을 체계적인 방식으로 실행하기 위한 마이그레이션 방법론과 일반적인 마이그레이션 시나리오를 자동화하고 가속화하는 도구 세트가 포함되어 있습니다.

대규모 마이그레이션

애플리케이션 포트폴리오의 대다수를 웨이브를 통해 클라우드로 이동하는 프로세스로, 각 웨이브에서 더 많은 애플리케이션이 더 빠른 속도로 이동합니다. 이 단계에서는 이전 단계에서 배운 모범 사례와 교훈을 사용하여 팀, 도구 및 프로세스의 마이그레이션 팩토리를 구현하여 자동화 및 민첩한 제공을 통해 워크로드 마이그레이션을 간소화합니다. 이것은 AWS 마이그레이션 전략의 세 번째 단계입니다.

마이그레이션 팩토리

자동화되고 민첩한 접근 방식을 통해 워크로드 마이그레이션을 간소화하는 다기능 팀입니다. 마이그레이션 팩토리 팀에는 일반적으로 스프린트에서 일하는 운영, 비즈니스 분석가 및 소유자, 마이그레이션 엔지니어, 개발자, DevOps 전문가가 포함됩니다. 엔터프라이즈 애플리케이션 포트폴리오의 20~50%는 공장 접근 방식으로 최적화할 수 있는 반복되는 패턴으로 구성되어 있습니다. 자세한 내용은 이 콘텐츠 세트의 클라우드 마이그레이션 팩토리 가이드와 마이그레이션 팩토리에 대한 설명을 참조하십시오.

마이그레이션 메타데이터

마이그레이션을 완료하는 데 필요한 애플리케이션 및 서버에 대한 정보 각 마이그레이션 패턴에는 서로 다른 마이그레이션 메타데이터 세트가 필요합니다. 마이그레이션 메타데이터의 예로는 대상 서브넷, 보안 그룹 및 AWS 계정이 있습니다.

마이그레이션 패턴

사용되는 마이그레이션 전략, 마이그레이션 대상, 마이그레이션 애플리케이션 또는 서비스를 자세히 설명하는 반복 가능한 마이그레이션 작업입니다. 예: AWS Application Migration Service를 사용하여 Amazon EC2로 마이그레이션을 리호스팅합니다.

Migration Portfolio Assessment(MPA)

AWS 클라우드로 마이그레이션하는 비즈니스 사례를 검증하기 위한 정보를 제공하는 온라인 도구입니다. MPA는 상세한 포트폴리오 평가(서버 적정 규모 조정, 가격 책정, TCO 비교, 마이그레이션 비용 분석)와 마이그레이션 계획(애플리케이션 데이터 분석 및 데이터 수집, 애플리케이션 그룹화, 마이그레이션 우선순위 지정, 웨이브 계획)을 제공합니다. MPA 도구(로그인 필요)는 모든 AWS 컨설턴트와 APN 파트너 컨설턴트가 무료로 사용할 수 있습니다.

마이그레이션 준비 상태 평가(MRA)

AWS CAF를 사용하여 조직의 클라우드 준비 상태에 대한 인사이트를 얻고, 강점과 약점을 식별하고, 식별된 격차를 해소하기 위한 행동 계획을 수립하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 가이드를 참조하십시오. MRA는 AWS 마이그레이션 전략의 첫 번째 단계입니다.

마이그레이션 전략

워크로드를 AWS 클라우드로 마이그레이션하는 데 사용되는 접근 방식입니다. 자세한 내용은 이 용어집의 7R 항목과 조직을 동원하여 대규모 마이그레이션 가속화를 참조하세요.

ML

기계 학습을 참조하세요.

현대화

비용을 절감하고 효율성을 높이고 혁신을 활용하기 위해 구식(레거시 또는 모놀리식) 애플리케이션과 해당 인프라를 클라우드의 민첩하고 탄력적이고 가용성이 높은 시스템으로 전환하는 것입니다. 자세한 내용은 AWS 클라우드에서 애플리케이션을 현대화하기 위한 전략을 참조하세요.

현대화 준비 상태 평가

조직 애플리케이션의 현대화 준비 상태를 파악하고, 이점, 위험 및 종속성을 식별하고, 조직이 해당 애플리케이션의 향후 상태를 얼마나 잘 지원할 수 있는지를 확인하는 데 도움이 되는 평가입니다. 평가 결과는 대상 아키텍처의 청사진, 현대화 프로세스의 개발 단계와 마일스톤을 자세히 설명하는 로드맵 및 파악된 격차를 해소하기 위한 실행 계획입니다. 자세한 내용은 AWS 클라우드에서 애플리케이션의 현대화 준비 상태 평가를 참조하세요.

모놀리식 애플리케이션(모놀리식 유형)

긴밀하게 연결된 프로세스를 사용하여 단일 서비스로 실행되는 애플리케이션입니다. 모놀리식 애플리케이션에는 몇 가지 단점이 있습니다. 한 애플리케이션 기능에 대한 수요가 급증하면 전체 아키텍처 규모를 조정해야 합니다. 코드 베이스가 커지면 모놀리식 애플리케이션의 기능을 추가하거나 개선하는 것도 더 복잡해집니다. 이러한 문제를 해결하기 위해 마이크로서비스 아키텍처를 사용할 수 있습니다. 자세한 내용은 마이크로서비스로 모놀리식 유형 분해를 참조하십시오.

MPA

Migration Portfolio Assessment를 참조하세요.

MQTT

메시지 큐 원격 분석 전송을 참조하세요.

멀티클래스 분류

여러 클래스에 대한 예측(2개 이상의 결과 중 하나 예측)을 생성하는 데 도움이 되는 프로세스입니다. 예를 들어, ML 모델이 '이 제품은 책인가요, 자동차인가요, 휴대폰인가요?' 또는 '이 고객이 가장 관심을 갖는 제품 범주는 무엇인가요?'라고 물을 수 있습니다.

변경 가능한 인프라

프로덕션 워크로드에 대한 기존 인프라를 업데이트하고 수정하는 모델입니다. 일관성, 신뢰성 및 예측 가능성을 높이기 위해 AWS Well-Architected Framework에서는 변경 불가능한 인프라를 모범 사례로 사용할 것을 권장합니다.

O

OAC

오리진 액세스 제어를 참조하세요.

OAI

오리진 액세스 ID를 참조하세요.

OCM

조직 변경 관리를 참조하세요.

오프라인 마이그레이션

마이그레이션 프로세스 중 소스 워크로드가 중단되는 마이그레이션 방법입니다. 이 방법은 가동 중지 증가를 수반하며 일반적으로 작고 중요하지 않은 워크로드에 사용됩니다.

OI

운영 통합을 참조하세요.

OLA

운영 수준 계약을 참조하세요.

온라인 마이그레이션

소스 워크로드를 오프라인 상태로 전환하지 않고 대상 시스템에 복사하는 마이그레이션 방법입니다. 워크로드에 연결된 애플리케이션은 마이그레이션 중에도 계속 작동할 수 있습니다. 이 방법은 가동 중지 차단 또는 최소화를 수반하며 일반적으로 중요한 프로덕션 워크로드에 사용됩니다.

OPC-UA

Open Process Communications - Unified Architecture(OPC-UA)를 참조하세요.

Open Process Communications - Unified Architecture(OPC-UA)

산업 자동화를 위한 Machine-to-Machine(M2M) 통신 프로토콜입니다. OPC-UA는 데이터 암호화, 인증 및 권한 부여 체계에 관한 상호 운용성 표준을 제공합니다.

운영 수준 협약(OLA)

서비스 수준에 관한 계약(SLA)을 지원하기 위해 직무 IT 그룹이 서로에게 제공하기로 약속한 내용을 명확히 하는 계약입니다.

운영 준비 상태 검토(ORR)

인시던트 및 잠재적 장애의 범위를 이해, 평가 또는 예방하거나 줄이는 데 도움이 되는 질문 체크리스트 및 관련 모범 사례입니다. 자세한 내용은 AWS Well-Architected Framework의 운영 준비 상태 검토(ORR)를 참조하세요.

운영 기술(OT)

물리적 환경에서 작동하여 산업 운영, 장비 및 인프라를 제어하는 하드웨어 및 소프트웨어 시스템입니다. 제조 분야에서 OT 및 정보 기술(IT) 시스템의 통합은 Industry 4.0 트랜스포메이션의 주요 중점 사항입니다.

운영 통합(OI)

클라우드에서 운영을 현대화하는 프로세스로 준비 계획, 자동화 및 통합을 수반합니다. 자세한 내용은 운영 통합 가이드를 참조하십시오.

조직 트레일

조직 AWS 계정 내 모든에 대한 모든 이벤트를 로깅 AWS CloudTrail 하는에서 생성된 추적입니다 AWS Organizations. 이 트레일은 조직에 속한 각 AWS 계정 에 생성되고 각 계정의 활동을 추적합니다. 자세한 내용은 CloudTrail 설명서의 Creating a trail for an organization을 참조하십시오.

조직 변경 관리(OCM)

사람, 문화 및 리더십 관점에서 중대하고 파괴적인 비즈니스 혁신을 관리하기 위한 프레임워크입니다. OCM은 변화 채택을 가속화하고, 과도기적 문제를 해결하고, 문화 및 조직적 변화를 주도함으로써 조직이 새로운 시스템 및 전략을 준비하고 전환할 수 있도록 지원합니다. AWS 마이그레이션 전략에서는 클라우드 채택 프로젝트에 필요한 변경 속도 때문에이 프레임워크를 인력 가속화라고 합니다. 자세한 내용은 사용 가이드를 참조하십시오.

오리진 액세스 제어(OAC)

CloudFront에서 Amazon Simple Storage Service(S3) 콘텐츠를 보호하기 위해 액세스를 제한하는 고급 옵션입니다. OAC는 AWS KMS (SSE-KMS)를 사용한 모든 서버 측 암호화 AWS 리전와 S3 버킷에 대한 동적 PUT 및 DELETE 요청에서 모든 S3 버킷을 지원합니다.

오리진 액세스 ID(OAI)

CloudFront에서 Amazon S3 콘텐츠를 보호하기 위해 액세스를 제한하는 옵션입니다. OAI를 사용하면 CloudFront는 Amazon S3가 인증할 수 있는 보안 주체를 생성합니다. 인증된 보안 주체는 특

정 CloudFront 배포를 통해서만 S3 버킷의 콘텐츠에 액세스할 수 있습니다. 더 세분화되고 향상된 액세스 제어를 제공하는 OAC도 참조하십시오.

ORR

운영 준비 상태 검토를 참조하세요.

OT

운영 기술을 참조하세요.

아웃바운드(송신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 내에서 시작된 네트워크 연결을 처리하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

P

권한 경계

사용자나 역할이 가질 수 있는 최대 권한을 설정하기 위해 IAM 보안 주체에 연결되는 IAM 관리 정책입니다. 자세한 내용은 IAM 설명서의 권한 경계를 참조하십시오.

개인 식별 정보(PII)

직접 보거나 다른 관련 데이터와 함께 짝을 지을 때 개인의 신원을 합리적으로 추론하는 데 사용할 수 있는 정보입니다. PII의 예로는 이름, 주소, 연락처 정보 등이 있습니다.

PII

개인 식별 정보를 참조하세요.

플레이북

클라우드에서 핵심 운영 기능을 제공하는 등 마이그레이션과 관련된 작업을 캡처하는 일련의 사전 정의된 단계입니다. 플레이북은 스크립트, 자동화된 런북 또는 현대화된 환경을 운영하는 데 필요한 프로세스나 단계 요약의 형태를 취할 수 있습니다.

PLC

프로그래밍 가능 로직 컨트롤러를 참조하세요.

PLM

제품 수명 주기 관리를 참조하세요.

정책

권한 정의(ID 기반 정책 참조), 액세스 조건 지정(리소스 기반 정책 참조), AWS Organizations 내 조직의 모든 계정에 대한 최대 권한 정의(서비스 제어 정책 참조)와 같은 작업을 수행할 수 있는 객체입니다.

다국어 지속성

데이터 액세스 패턴 및 기타 요구 사항을 기반으로 독립적으로 마이크로서비스의 데이터 스토리지 기술 선택. 마이크로서비스가 동일한 데이터 스토리지 기술을 사용하는 경우 구현 문제가 발생하거나 성능이 저하될 수 있습니다. 요구 사항에 가장 적합한 데이터 저장소를 사용하면 마이크로서비스를 더 쉽게 구현하고 성능과 확장성을 높일 수 있습니다.

포트폴리오 평가

마이그레이션을 계획하기 위해 애플리케이션 포트폴리오를 검색 및 분석하고 우선순위를 정하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 상태 평가를 참조하십시오.

조건자

보통 `WHERE` 절에 있는 `true` 또는 `false`를 반환하는 쿼리 조건입니다.

푸시다운 조건자

전송 전에 쿼리의 데이터를 필터링하는 데이터베이스 쿼리 최적화 기법입니다. 이렇게 하면 관계형 데이터베이스에서 검색하고 처리해야 하는 데이터의 양이 줄고 쿼리 성능이 향상됩니다.

예방적 제어

이벤트 발생을 방지하도록 설계된 보안 제어입니다. 이 제어는 네트워크에 대한 무단 액세스나 원치 않는 변경을 방지하는 데 도움이 되는 1차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Preventative controls를 참조하십시오.

보안 주체

작업을 수행하고 리소스에 액세스할 수 AWS 있는의 엔터티입니다. 이 엔터티는 일반적으로 , AWS 계정 IAM 역할 또는 사용자의 루트 사용자입니다. 자세한 내용은 IAM 설명서의 역할 용어 및 개념의 보안 주체를 참조하십시오.

개인 정보 보호 중심 설계

전체 개발 프로세스에서 개인 정보를 고려하는 시스템 엔지니어링에서의 접근 방식입니다.

프라이빗 호스팅 영역

Amazon Route 53에서 하나 이상의 VPC 내 도메인과 하위 도메인에 대한 DNS 쿼리에 응답하는 방법에 대한 정보가 담긴 컨테이너입니다. 자세한 내용은 Route 53 설명서의 프라이빗 호스팅 영역 작업을 참조하십시오.

선제적 제어

규정 미준수 리소스의 배포를 방지하도록 설계된 보안 제어입니다. 이러한 제어는 리소스를 프로비저닝하기 전에 리소스를 스캔합니다. 리소스가 제어를 준수하지 않으면 프로비저닝되지 않습니다. 자세한 내용은 AWS Control Tower 설명서의 제어 참조 가이드를 참조하고 보안 제어 구현의 사전 예방적 제어를 참조하세요. AWS

제품 수명 주기 관리(PLM)

설계, 개발 및 출시부터 성장 및 성숙도를 거쳐 거부 및 제거에 이르기까지 전체 수명 주기 동안 제품의 데이터 및 프로세스 관리를 나타냅니다.

프로덕션 환경

환경을 참조하세요.

프로그래밍 가능 로직 컨트롤러(PLC)

제조 분야에서 기계를 모니터링하고 제조 프로세스를 자동화하는 매우 안정적이고 적응력이 뛰어난 컴퓨터입니다.

프롬프트 체이닝

한 LLM 프롬프트의 출력을 다음 프롬프트의 입력으로 사용하여 더 나은 응답을 생성합니다. 이 기법은 복잡한 태스크를 하위 태스크로 나누거나 예비 응답을 반복적으로 세부 조정하거나 확장하는 데 사용됩니다. 이를 통해 모델 응답의 정확성과 관련성을 개선하고 보다 세분화되고 개인화된 결과를 얻을 수 있습니다.

가명화

데이터세트의 개인 식별자를 자리 표시자 값으로 바꾸는 프로세스입니다. 가명화는 개인 정보를 보호하는 데 도움이 될 수 있습니다. 가명화된 데이터는 여전히 개인 데이터로 간주됩니다.

게시/구독(pub/sub)

여러 마이크로서비스에서 비동기 통신을 지원하여 확장성과 응답성을 개선하는 패턴입니다. 예를 들어 마이크로서비스 기반 MES에서 마이크로서비스는 다른 마이크로서비스가 구독할 수 있는 채널에 이벤트 메시지를 게시할 수 있습니다. 시스템은 게시 서비스를 변경하지 않고도 새 마이크로서비스를 추가할 수 있습니다.

Q

쿼리 계획

SQL 관계형 데이터베이스 시스템의 데이터에 액세스하는 데 사용되는 명령어와 같은 일련의 단계입니다.

쿼리 계획 회귀

데이터베이스 서비스 최적화 프로그램이 데이터베이스 환경을 변경하기 전보다 덜 최적의 계획을 선택하는 경우입니다. 통계, 제한 사항, 환경 설정, 쿼리 파라미터 바인딩 및 데이터베이스 엔진 업데이트의 변경으로 인해 발생할 수 있습니다.

R

RACI 매트릭스

Responsible, Accountable, Consulted, Informed(RACI)를 참조하세요.

RAG

검색 증강 생성을 참조하세요.

랜섬웨어

결제가 완료될 때까지 컴퓨터 시스템이나 데이터에 대한 액세스를 차단하도록 설계된 악성 소프트웨어입니다.

RASCI 매트릭스

Responsible, Accountable, Consulted, Informed(RACI)를 참조하세요.

RCAC

행 및 열 액세스 제어를 참조하세요.

읽기 전용 복제본

읽기 전용 용도로 사용되는 데이터베이스의 사본입니다. 쿼리를 읽기 전용 복제본으로 라우팅하여 기본 데이터베이스의 로드를 줄일 수 있습니다.

리아키텍팅

7R을 참조하세요.

Recovery Point Objective(RPO)

마지막 데이터 복구 시점 이후 허용되는 최대 시간입니다. 이에 따라 마지막 복구 시점과 서비스 중단 사이에 허용되는 데이터 손실로 간주되는 범위가 결정됩니다.

Recovery Time Objective(RTO)

서비스 중단과 서비스 복원 사이의 허용 가능한 지연 시간입니다.

리팩터링

7R을 참조하세요.

리전

지리적 영역의 AWS 리소스 모음입니다. 각 AWS 리전 는 내결함성, 안정성 및 복원력을 제공하기 위해 서로 격리되고 독립적입니다. 자세한 내용은 계정에서 사용할 수 있는 AWS 리전 지정을 참조하세요.

회귀

숫자 값을 예측하는 ML 기법입니다. 예를 들어, '이 집은 얼마에 팔릴까?'라는 문제를 풀기 위해 ML 모델은 선형 회귀 모델을 사용하여 주택에 대해 알려진 사실(예: 면적)을 기반으로 주택의 매매 가격을 예측할 수 있습니다.

리호스팅

7R을 참조하세요.

릴리스

배포 프로세스에서 변경 사항을 프로덕션 환경으로 승격시키는 행위입니다.

재배치

7R을 참조하세요.

리플랫포밍

7R을 참조하세요.

재구매

7R을 참조하세요.

복원력

중단에 저항하거나 중단을 복구할 수 있는 애플리케이션의 기능입니다. 고가용성 및 재해 복구는 AWS 클라우드에서 복원력을 계획할 때 일반적인 고려 사항입니다. 자세한 내용은 AWS 클라우드 복원력을 참조하세요.

리소스 기반 정책

Amazon S3 버킷, 엔드포인트, 암호화 키 등의 리소스에 연결된 정책입니다. 이 유형의 정책은 액세스가 허용된 보안 주체, 지원되는 작업 및 충족해야 하는 기타 조건을 지정합니다.

RACI(Responsible, Accountable, Consulted, Informed) 매트릭스

마이그레이션 활동 및 클라우드 운영에 참여하는 모든 당사자의 역할과 책임을 정의하는 매트릭스입니다. 매트릭스 이름은 매트릭스에 정의된 책임 유형에서 파생됩니다. 실무 담당자 (R), 의사 결정권자 (A), 업무 수행 조언자 (C), 결과 통보 대상자 (I). 지원자는 (S) 선택사항입니다. 지원자를 포함하면 매트릭스를 RASCI 매트릭스라고 하고, 지원자를 제외하면 RACI 매트릭스라고 합니다.

대응 제어

보안 기준에서 벗어나거나 부정적인 이벤트를 해결하도록 설계된 보안 제어입니다. 자세한 내용은 AWS에서 보안 제어 구현의 대응 제어를 참조하세요.

retain

7R을 참조하세요.

사용 중지

7R을 참조하세요.

검색 증강 세대(RAG)

응답을 생성하기 전에 LLM이 훈련 데이터 소스 외부에 있는 신뢰할 수 있는 데이터 소스를 참조하는 생성형 AI 기술입니다. 예를 들어 RAG 모델은 조직의 지식 기반 또는 사용자 지정 데이터에 대한 시맨틱 검색을 수행할 수 있습니다. 자세한 내용은 검색 증강 생성(RAG)이란 무엇인가요?를 참조하세요.

교체

공격자가 자격 증명에 액세스하는 것을 더욱 어렵게 만들기 위해 보안 암호를 주기적으로 업데이트하는 프로세스입니다.

행 및 열 액세스 제어(RCAC)

액세스 규칙이 정의된 기본적이고 유연한 SQL 표현식을 사용합니다. RCAC는 행 권한과 열 마스크로 구성됩니다.

RPO

목표 복구 시점(RPO)을 참조하세요.

RTO

목표 복구 시간(RTO)을 참조하세요.

런북

특정 작업을 수행하는 데 필요한 일련의 수동 또는 자동 절차입니다. 일반적으로 오류율이 높은 반복 작업이나 절차를 간소화하기 위해 런북을 만듭니다.

S

SAML 2.0

많은 ID 제공업체(idP)에서 사용하는 개방형 표준입니다. 이 기능을 사용하면 연동 SSO(Single Sign-On)를 AWS Management Console 사용할 수 있으므로 사용자는 조직의 모든 사용자에 대해 IAM에서 사용자를 생성하지 않고도에 로그인하거나 AWS API 작업을 호출할 수 있습니다. SAML 2.0 기반 페더레이션에 대한 자세한 내용은 IAM 설명서의 SAML 2.0 기반 페더레이션 정보를 참조하십시오.

SCADA

감독 제어 및 데이터 획득을 참조하세요.

SCP

서비스 제어 정책을 참조하세요.

보안 암호

에는 암호화된 형식으로 저장하는 암호 또는 사용자 자격 증명과 같은 AWS Secrets Manager기밀 또는 제한된 정보가 있습니다. 보안 암호 값과 메타데이터로 구성됩니다. 보안 암호 값은 바이너리, 단일 문자열 또는 여러 문자열일 수 있습니다. 자세한 내용은 AWS Secrets Manager 설명서의 Secrets Manager 보안 암호란 무엇인가요?를 참조하세요.

보안 중심 설계

전체 개발 프로세스에서 보안을 고려하는 시스템 엔지니어링에서의 접근 방식입니다.

보안 제어

위협 행위자가 보안 취약성을 악용하는 능력을 방지, 탐지 또는 감소시키는 기술적 또는 관리적 가드레일입니다. 보안 제어는 예방, 감지, 대응, 선제적과 같은 기본적인 네 가지 보안 제어 유형으로 구분됩니다.

보안 강화

공격 표면을 줄여 공격에 대한 저항력을 높이는 프로세스입니다. 더 이상 필요하지 않은 리소스 제거, 최소 권한 부여의 보안 모범 사례 구현, 구성 파일의 불필요한 기능 비활성화 등의 작업이 여기에 포함될 수 있습니다.

보안 정보 및 이벤트 관리(SIEM) 시스템

보안 정보 관리(SIM)와 보안 이벤트 관리(SEM) 시스템을 결합하는 도구 및 서비스입니다. SIEM 시스템은 서버, 네트워크, 디바이스 및 기타 소스에서 데이터를 수집, 모니터링 및 분석하여 위협과 보안 침해를 탐지하고 알림을 생성합니다.

보안 응답 자동화

보안 이벤트에 자동으로 응답하거나 이를 해결하도록 설계된 사전 정의되고 프로그래밍된 작업입니다. 이러한 자동화는 보안 모범 사례를 구현하는 데 도움이 되는 탐지 또는 대응 AWS 보안 제어 역할을 합니다. 자동화된 응답 작업의 예로 VPC 보안 그룹 수정, Amazon EC2 인스턴스 패치 적용 또는 자격 증명 교체 등이 있습니다.

서버 측 암호화

대상에서 데이터를 수신하는 AWS 서비스 에 의한 데이터 암호화.

서비스 제어 정책(SCP)

AWS Organizations에 속한 조직의 모든 계정에 대한 권한을 중앙 집중식으로 제어하는 정책입니다. SCP는 관리자가 사용자 또는 역할에 위임할 수 있는 작업에 대해 제한을 설정하거나 가드레일을 정의합니다. SCP를 허용 목록 또는 거부 목록으로 사용하여 허용하거나 금지할 서비스 또는 작업을 지정할 수 있습니다. 자세한 내용은 AWS Organizations 설명서의 서비스 제어 정책을 참조하세요.

서비스 엔드포인트

에 대한 진입점의 URL입니다 AWS 서비스. 엔드포인트를 사용하여 대상 서비스에 프로그래밍 방식으로 연결할 수 있습니다. 자세한 내용은 AWS 일반 참조의 AWS 서비스 엔드포인트를 참조하십시오.

서비스 수준에 관한 계약(SLA)

IT 팀이 고객에게 제공하기로 약속한 내용(예: 서비스 가동 시간 및 성능)을 명시한 계약입니다.

서비스 수준 지표(SLI)

오류 발생률, 가용성 또는 처리량과 같은 서비스의 성능 측면에 대한 측정값입니다.

서비스 수준 목표(SLO)

서비스 수준 지표로 측정되는 서비스의 상태를 나타내는 목표 지표입니다.

공동 책임 모델

클라우드 보안 및 규정 준수를 AWS 위해와 공유하는 책임을 설명하는 모델입니다. AWS 는 클라우드의 보안을 담당하는 반면, 사용자는 클라우드의 보안을 담당합니다. 자세한 내용은 공동 책임 모델을 참조하십시오.

SIEM

보안 정보 및 이벤트 관리 시스템을 참조하세요.

단일 장애점(SPOF)

애플리케이션을 중단시킬 수 있는 애플리케이션의 중요한 단일 구성 요소에서 발생하는 장애입니다.

SLA

서비스 수준 계약을 참조하세요.

SLI

서비스 수준 지표를 참조하세요.

SLO

서비스 수준 목표를 참조하세요.

분할 앤 시드 모델

현대화 프로젝트를 확장하고 가속화하기 위한 패턴입니다. 새로운 기능과 제품 릴리스가 정의되면 핵심 팀이 분할되어 새로운 제품 팀이 만들어집니다. 이를 통해 조직의 역량과 서비스 규모를 조정하고, 개발자 생산성을 개선하고, 신속한 혁신을 지원할 수 있습니다. 자세한 내용은 AWS 클라우드에서 애플리케이션을 현대화하기 위한 단계별 접근 방식을 참조하세요.

SPOF

단일 장애점을 참조하세요.

스타 스키마

하나의 큰 팩트 테이블을 사용하여 트랜잭션 또는 측정된 데이터를 저장하고 하나 이상의 더 작은 차원 테이블을 사용하여 데이터 속성을 저장하는 데이터베이스 조직 구조입니다. 이 구조는 데이터 웨어하우스에서 또는 비즈니스 인텔리전스 목적으로 사용하도록 설계되었습니다.

Strangler Fig 패턴

레거시 시스템을 폐기할 수 있을 때까지 시스템 기능을 점진적으로 다시 작성하고 교체하여 모놀리식 시스템을 현대화하기 위한 접근 방식. 이 패턴은 무화과 덩굴이 나무로 자라 결국 숙주를 압도하고 대체하는 것과 비슷합니다. Martin Fowler가 모놀리식 시스템을 다시 작성할 때 위험을 관리하는 방법으로 이 패턴을 도입했습니다. 이 패턴을 적용하는 방법의 예는 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

서브넷

VPC의 IP 주소 범위입니다. 서브넷은 단일 가용 영역에 상주해야 합니다.

감독 제어 및 데이터 획득(SCADA)

제조 분야에서 하드웨어와 소프트웨어를 사용하여 물리적 자산과 프로덕션 작업을 모니터링하는 시스템입니다.

대칭 암호화

동일한 키를 사용하여 데이터를 암호화하고 복호화하는 암호화 알고리즘입니다.

합성 테스트

사용자 상호 작용을 시뮬레이션하여 잠재적 문제를 감지하거나 성능을 모니터링하는 방식으로 진행되는 시스템 테스트입니다. Amazon CloudWatch Synthetics를 사용하여 이러한 테스트를 생성할 수 있습니다.

시스템 프롬프트

LLM에 컨텍스트, 명령 또는 지침을 제공하여 동작을 지시하는 기법입니다. 시스템 프롬프트는 컨텍스트를 설정하고 사용자와의 상호 작용을 위한 규칙을 설정하는 데 도움이 됩니다.

T

tags

AWS 리소스를 구성하기 위한 메타데이터 역할을 하는 키-값 페어입니다. 태그를 사용하면 리소스를 손쉽게 관리, 식별, 정리, 검색, 필터링할 수 있습니다. 자세한 내용은 AWS 리소스에 태그 지정을 참조하십시오.

대상 변수

지도 ML에서 예측하려는 값으로, 결과 변수라고도 합니다. 예를 들어, 제조 설정에서 대상 변수는 제품 결함일 수 있습니다.

작업 목록

런북을 통해 진행 상황을 추적하는 데 사용되는 도구입니다. 작업 목록에는 런북의 개요와 완료해야 할 일반 작업 목록이 포함되어 있습니다. 각 일반 작업에 대한 예상 소요 시간, 소유자 및 진행 상황이 작업 목록에 포함됩니다.

테스트 환경

환경을 참조하세요.

훈련

ML 모델이 학습할 수 있는 데이터를 제공하는 것입니다. 훈련 데이터에는 정답이 포함되어야 합니다. 학습 알고리즘은 훈련 데이터에서 대상(예측하려는 답)에 입력 데이터 속성을 매핑하는 패턴을 찾고, 이러한 패턴을 캡처하는 ML 모델을 출력합니다. 그런 다음 ML 모델을 사용하여 대상을 모르는 새 데이터에 대한 예측을 할 수 있습니다.

Transit Gateway

VPC와 온프레미스 네트워크를 상호 연결하는 데 사용할 수 있는 네트워크 전송 허브입니다. 자세한 내용은 AWS Transit Gateway 설명서의 전송 게이트웨이란 무엇입니까?를 참조하세요.

트렁크 기반 워크플로

개발자가 기능 브랜치에서 로컬로 기능을 구축하고 테스트한 다음 해당 변경 사항을 기본 브랜치에 병합하는 접근 방식입니다. 이후 기본 브랜치는 개발, 프로덕션 이전 및 프로덕션 환경에 순차적으로 구축됩니다.

신뢰할 수 있는 액세스

사용자를 대신하여 AWS Organizations 및 해당 계정에서 조직에서 작업을 수행하도록 지정하는 서비스에 대한 권한 부여. 신뢰할 수 있는 서비스는 필요할 때 각 계정에 서비스 연결 역할을 생성하여 관리 작업을 수행합니다. 자세한 내용은 설명서의 다른 AWS 서비스와 AWS Organizations 함께 사용을 참조하세요 AWS Organizations .

튜닝

ML 모델의 정확도를 높이기 위해 훈련 프로세스의 측면을 여러 변경하는 것입니다. 예를 들어, 레이블링 세트를 생성하고 레이블을 추가한 다음 다양한 설정에서 이러한 단계를 여러 번 반복하여 모델을 최적화하는 방식으로 ML 모델을 훈련할 수 있습니다.

피자 두 판 팀

피자 두 판이면 충분한 소규모 DevOps 팀. 피자 두 판 팀 규모는 소프트웨어 개발에 있어 가능한 최상의 공동 작업 기회를 보장합니다.

U

불확실성

예측 ML 모델의 신뢰성을 저해할 수 있는 부정확하거나 불완전하거나 알려지지 않은 정보를 나타내는 개념입니다. 불확실성에는 두 가지 유형이 있습니다. 인식론적 불확실성은 제한적이고 불완전한 데이터에 의해 발생하는 반면, 우연한 불확실성은 데이터에 내재된 노이즈와 무작위성에 의해 발생합니다. 자세한 내용은 Quantifying uncertainty in deep learning systems 가이드를 참조하십시오.

차별화되지 않은 작업

애플리케이션을 만들고 운영하는 데 필요하지만 최종 사용자에게 직접적인 가치를 제공하거나 경쟁 우위를 제공하지 못하는 작업을 헤비 리프팅이라고도 합니다. 차별화되지 않은 작업의 예로는 조달, 유지보수, 용량 계획 등이 있습니다.

상위 환경

환경을 참조하세요.

V

정리

스토리지를 회수하고 성능을 향상시키기 위해 증분 업데이트 후 정리 작업을 수반하는 데이터베이스 유지 관리 작업입니다.

버전 제어

리포지토리의 소스 코드 변경과 같은 변경 사항을 추적하는 프로세스 및 도구입니다.

VPC 피어링

프라이빗 IP 주소를 사용하여 트래픽을 라우팅할 수 있게 하는 두 VPC 간의 연결입니다. 자세한 내용은 Amazon VPC 설명서의 VPC 피어링이란?을 참조하십시오.

취약성

시스템 보안을 손상시키는 소프트웨어 또는 하드웨어 결함입니다.

W

웜 캐시

자주 액세스하는 최신 관련 데이터를 포함하는 버퍼 캐시입니다. 버퍼 캐시에서 데이터베이스 인스턴스를 읽을 수 있기 때문에 주 메모리나 디스크에서 읽는 것보다 빠릅니다.

웜 데이터

자주 액세스하지 않는 데이터입니다. 이런 종류의 데이터를 쿼리할 때는 일반적으로 적절히 느린 쿼리가 허용됩니다.

창 함수

현재 레코드와 어떤 식으로든 관련된 행 그룹에서 계산을 수행하는 SQL 함수입니다. 창 함수는 이동 평균을 계산하거나 현재 행의 상대적 위치를 기반으로 행 값에 액세스하는 등의 태스크를 처리하는 데 유용합니다.

워크로드

고객 대면 애플리케이션이나 백엔드 프로세스 같이 비즈니스 가치를 창출하는 리소스 및 코드 모음입니다.

워크스트림

마이그레이션 프로젝트에서 특정 작업 세트를 담당하는 직무 그룹입니다. 각 워크스트림은 독립적이지만 프로젝트의 다른 워크스트림을 지원합니다. 예를 들어, 포트폴리오 워크스트림은 애플리케이션 우선순위 지정, 웨이브 계획, 마이그레이션 메타데이터 수집을 담당합니다. 포트폴리오 워크스트림은 이러한 자산을 마이그레이션 워크스트림에 전달하고, 마이그레이션 워크스트림은 서버와 애플리케이션을 마이그레이션합니다.

WORM

Write Once, Read Many(WORM)를 참조하세요.

WQF

AWS Workload Qualification Framework를 참조하세요.

Write Once Read Many(WORM)

데이터를 한 번 쓰고 데이터가 삭제되거나 수정되지 않도록 하는 스토리지 모델입니다. 권한 있는 사용자는 필요한 만큼 여러 번 데이터를 읽을 수 있지만 데이터를 변경할 수는 없습니다. 이 데이터 스토리지 인프라는 변경 불가능한 항목으로 간주됩니다.

Z

제로데이 익스플로잇

제로데이 취약성을 악용하는 공격(일반적으로 맬웨어)입니다.

제로데이 취약성

프로덕션 시스템의 명백한 결함 또는 취약성입니다. 위협 행위자는 이러한 유형의 취약성을 사용하여 시스템을 공격할 수 있습니다. 개발자는 공격의 결과로 취약성을 인지하는 경우가 많습니다.

제로샷 프롬프팅

태스크를 수행하기 위해 LLM에 명령을 제공하지만 안내에 도움이 되는 예제(샷)는 제공하지 않습니다. LLM은 사전 훈련된 지식을 사용하여 태스크를 처리해야 합니다. 제로샷 프롬프팅의 효과는 태스크의 복잡성과 프롬프트의 품질에 따라 달라집니다. 퓨샷 프롬프팅도 참조하세요.

좀비 애플리케이션

평균 CPU 및 메모리 사용량이 5% 미만인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하는 것이 일반적입니다.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.